Approximability of Monotone Submodular Function Maximization under Cardinality and Matroid Constraints in the Streaming Model

Chien-Chung Huang, Naonori Kakimura, Simon Mauras, Yuichi Yoshida

Approximability of Monotone Submodular Function Maximization under Cardinality and Matroid Constraints in the Streaming Model

Chien-Chung Huang
CNRS, DI ENS, PSL
villars@gmail.com

Naonori Kakimura*
Keio University
kakimura@math.keio.ac.jp

Simon Mauras
Université de Paris, IRIF, CNRS
simon.mauras@irif.fr

Yuichi Yoshida
National Institute of Informatics
yyoshida@nii.ac.jp

February 14, 2020

Abstract

Maximizing a monotone submodular function under various constraints is a classical and intensively studied problem. However, in the single-pass streaming model, where the elements arrive one by one and an algorithm can store only a small fraction of input elements, there is much gap in our knowledge, even though several approximation algorithms have been proposed in the literature.

In this work, we present the first lower bound on the approximation ratios for cardinality and matroid constraints that beat $1 - \frac{1}{e}$ in the single-pass streaming model. Let n be the number of elements in the stream. Then, we prove that any (randomized) streaming algorithm for a cardinality constraint with approximation ratio $\frac{2}{2+\sqrt{2}} + \varepsilon$ requires $\Omega\left(\frac{n}{K^2}\right)$ space for any $\varepsilon > 0$, where K is the size limit of the output set. We also prove that any (randomized) streaming algorithm for a (partition) matroid constraint with approximation ratio $\frac{K}{2K-1} + \varepsilon$ requires $\Omega\left(\frac{n}{K}\right)$ space for any $\varepsilon > 0$, where K is the rank of the given matroid.

In addition, we give streaming algorithms when we only have a weak oracle with which we can only evaluate function values on feasible sets. Specifically, we show weak-oracle streaming algorithms for cardinality and matroid constraints with approximation ratios $\frac{K}{2K-1}$ and $\frac{1}{2}$, respectively, whose space complexity is exponential in K but is independent of n. The former one exactly matches the known inapproximability result for a cardinality constraint in the weak oracle model. The latter one almost matches our lower bound of $\frac{K}{2K-1}$ for a matroid constraint, which almost settles the approximation ratio for a matroid constraint that can be obtained by a streaming algorithm whose space complexity is independent of n.

*Supported by JST ERATO Grant Number JPMJER1201, Japan, and by JSPS KAKENHI Grant Number JP17K00028.

1 Introduction

A set function $f\colon 2^E \to \mathbb{R}$ on a ground set E is *submodular* if it satisfies the *diminishing marginal return property*, i.e., for any subsets $S \subseteq T \subsetneq E$ and $e \in E \setminus T$,

$$f(S \cup \{e\}) - f(S) \geq f(T \cup \{e\}) - f(T).$$

A function is *monotone* if $f(S) \leq f(T)$ for any $S \subseteq T \subseteq E$. Submodular functions play a fundamental role in combinatorial optimization, as they capture rank functions of matroids, edge cuts of graphs, and set coverage, just to name a few examples.

In addition to their theoretical interests, submodular functions have attracted much attention from the machine learning community because they can model various practical problems such as online advertising [1, 27, 38], sensor location [28], text summarization [33, 34], and maximum entropy sampling [31]. Many of these problems can be formulated as non-negative monotone submodular function maximization under a cardinality constraint or a matroid constraint. Namely,

$$\text{(Cardinality constraint)} \quad \text{maximize } f(S) \quad \text{subject to } |S| \leq K, \quad S \subseteq E. \tag{1}$$

$$\text{(Matroid constraint)} \quad \text{maximize } f(S) \quad \text{subject to } S \in \mathcal{I}, \quad S \subseteq E, \tag{2}$$

where $f\colon 2^E \to \mathbb{R}_+$ is a monotone submodular function, $K \in \mathbb{Z}_+$ is a non-negative integer, and $\mathcal{M} = (E, \mathcal{I})$ is a matroid with independent family $\mathcal{I}$. Note that a matroid constraint includes a cardinality constraint as a special case: Choose the matroid in (2) to be the uniform matroid of rank K.

In some applications mentioned before, the amount of input data is much larger than the main memory capacity of individual computers. Then, it is natural to consider the *streaming model*, where each item in the ground set E arrives sequentially, and we are allowed to use a small amount of memory. Unless stated otherwise, we always consider *single-pass algorithms*, that is, algorithms that scan the entire stream only once.

Submodular maximization under the streaming model has received much attention recently. Algorithms with various approximation ratios and space requirements have been proposed for the cardinality constraint [2, 26], the knapsack constraint [24, 25, 42], and the matroid constraint [10, 13]. However, there are only a few inapproximability results. McGregor and Vu [35] showed that any streaming algorithm for maximizing a coverage function under a cardinality constraint with approximation ratio better than $1 - \frac{1}{e}$ requires $\Omega\left(\frac{n}{K^2}\right)$ space, where $n := |E|$ is the number of elements. Norouzi-Fard et al. [37] showed that any streaming algorithm for maximizing a monotone submodular function under a cardinality constraint with approximation ratio better than $\frac{K}{2K-1}$ requires $\Omega\left(\frac{n}{K}\right)$ space, assuming that we can only evaluate function values of feasible sets, which we call the *weak oracle model*. A standard value oracle is called *strong* for comparison.

1.1 Our contributions

The first contribution of this work is giving inapproximability results for cardinality and matroid constraints that beat $1 - \frac{1}{e}$ in the strong oracle model for the first time.

Table 1: Summary of results

Constraint		Approximation ratio	Space usage	Oracle	Reference
Cardinality	Algorithm	$\frac{1}{2}-\varepsilon$	$O\left(\frac{K}{\varepsilon}\right)$	weak	[26]
		$\frac{K}{2K-1}-\varepsilon$	$\widetilde{O}\left(\frac{K2^{2K}}{\varepsilon}\right)$	weak	Theorem 1.4
	Hardness	$1-\left(1-\frac{1}{K}\right)^K+\varepsilon, \forall \varepsilon>0$	$\Omega\left(\frac{n}{K^2}\right)$	strong	[35]
		$\frac{K}{2K-1}+\varepsilon, \forall \varepsilon>0$	$\Omega\left(\frac{n}{K}\right)$	weak	[37]
		$\frac{2}{2+\sqrt{2}}+\varepsilon, \forall \varepsilon>0$	$\Omega\left(\frac{n}{K^2}\right)$	strong	Theorem 1.2
Matroid	Algorithm	$\frac{1}{4}$	$K\log^{O(1)} n$	strong	[10, 13]
		$\frac{1}{2}-\varepsilon$	$\widetilde{O}\left(\frac{K^{5K+1}}{\varepsilon}\right)$	weak	Theorem 1.5
	Hardness	$\frac{K}{2K-1}+\varepsilon, \forall \varepsilon>0$	$\Omega\left(\frac{n}{K}\right)$	strong	Theorem 1.3

Before explaining our results, we first note that, in the context of submodular maximization, it is standard to assume that a value oracle of a function f is given and the complexity of algorithms is measured based on the number of oracle calls [2, 10, 13, 24, 25, 26, 42]. However, a value oracle of f is too powerful in the streaming setting if we are allowed to put an exponential number of queries. In fact, if we have a free access to the value oracle, we can maximize f even without seeing the stream by querying about every subset. This observation leads to the following natural model, which we call the *element-store model*.

Definition 1.1 (Element-store model)**.** *Let $E=\{e_1,\dots,e_n\}$ be the ground set and $f\colon 2^E\to\mathbb{R}_+$ be a set function. A streaming algorithm in the* element-store model *maintains a set of elements S, which is initially an empty set, and possess an additional memory M. At step $t\in\{1,\dots,n\}$, the item e_t is given to the algorithm, and the algorithm updates S and the content of M using the values of $f(S')$ for $S'\subseteq S\cup\{e_t\}$ and the content of M. Finally, the algorithm outputs a subset of S. The space complexity of the algorithm is the sum of the number of words stored in M and the maximum size of S over the n steps.*

The weak oracle model is equivalent to constraining S always to be a feasible set. We note that all known streaming algorithms for submodular function maximization [2, 10, 13, 24, 25, 26, 42] lie in the element-store model. Now, we state our results.

Theorem 1.2. *For any $K\in\mathbb{N}$ and $\varepsilon>0$, any (randomized) streaming algorithm for monotone submodular function maximization under a cardinality constraint in the element-store model with approximation ratio $\frac{2}{2+\sqrt{2}}+\varepsilon\approx 0.585+\varepsilon$ requires $\Omega\left(\frac{n}{K^2}\right)$ space.*

Theorem 1.3. *For any $K\in\mathbb{N}$ and $\varepsilon>0$, any (randomized) streaming algorithm for monotone submodular function maximization under a partition matroid constraint in the element-store model with approximation ratio $\frac{K}{2K-1}+\varepsilon$ requires $\Omega\left(\frac{n}{K}\right)$ space.*

Indeed, the same inapproximability results hold for any streaming algorithm in the element-store model with unbounded computational power and memory space for M, as long as the number

of elements stored in S is bounded. The proof techniques can be found in Section 1.2.

Next, we complement the previous and obtained inapproximability results by showing (weak-oracle) streaming algorithms for cardinality and matroid constraints. We first present a weak-oracle streaming algorithm for a cardinality constraint with approximation ratio $\frac{K}{2K-1} - \varepsilon$, which is slightly better than the previous best approximation ratio of $\frac{1}{2}$ [2, 26] and exactly matches the known inapproximability for the weak oracle model [37]. Although the space usage is exponential in K, it does not depend on the number of elements n.

Theorem 1.4. *There exists a weak-oracle $\left(\frac{K}{2K-1} - \varepsilon\right)$-approximation streaming algorithm for monotone submodular function maximization under a cardinality constraint with $O\left(\frac{K2^{2K}\log(K/\varepsilon)}{\varepsilon}\right)$ space.*

Then, we extend the algorithm given in Theorem 1.4 to a weak-oracle streaming algorithm for a matroid constraint with approximation ratio $\frac{1}{2} - \varepsilon$, which almost matches the inapproximability of $\frac{K}{2K-1} + \varepsilon$ given in Theorem 1.3. This almost settles the the approximation ratio for a matroid constraint that can be achieved by a streaming algorithm with space complexity independent of n, for both the weak and strong oracle models.

Theorem 1.5. *There exists a weak-oracle $\left(\frac{1}{2} - \varepsilon\right)$-approximation streaming algorithm for monotone submodular function maximization under a matroid constraint with $O\left(\frac{K^{5K+1}\log(K/\varepsilon)}{\varepsilon}\right)$ space.*

All the previous and obtained results are summarized in Table 1. Here, $\widetilde{O}(\cdot)$ hides a polylogarithmic factor in $\frac{K}{\varepsilon}$.

1.2 Our techniques

Lower bound construction We first describe the intuition behind our proof of Theorem 1.2. The elements of the ground set E are colored either blue, red, or purple, and we have a large number of $n - K$ blue elements, $K - 1$ red elements, and one purple element. Let B, R, and P be the set of blue, red, and purple elements, respectively, that is, $|B| = n - K$, $|R| = K - 1$, and $|P| = 1$. We note that the colors of elements are not revealed to algorithms. We design our monotone submodular function $f\colon 2^E \to \mathbb{Z}_+$ so that it is *colorwise-symmetric* meaning that the value of $f(S)$ is uniquely determined by the number of blue, red, and purple elements in a subset S. We write $f(b, r, p)$ to denote the value of $f(S)$ when there are b blue elements, r red elements, and p purple elements in S. We will assume that $f(0, K-1, 1)$ gives the optimal value.

In the input stream, blue and red elements arrive in a random order, and then the purple element arrives at the end. We design f so that it is hard to distinguish blue and red elements (without using the purple element). More precisely, f satisfies the property $f(b+1, 0, 0) = f(b, 1, 0)$ for any non-negative integer b. As the number of blue elements is much larger than that of red elements and the space is limited, with high probability, we must immediately throw away red elements from the memory right after they arrive. Thus, with high probability, we obtain the values $f(K, 0, 0)$, $f(K-1, 1, 0)$ or $f(K-1, 0, 1)$, i.e., most of the time the algorithm ends up with at least $K - 1$

blue elements. On the basis of some ideas suggested by computer simulations, we construct f so that the maximum of the three values is small.

The proof outline of Theorem 1.3 is similar, but is more involved. We first regard that the ground set E is partitioned into classes $C_1, \dots, C_K$ such that $|C_1| = \cdots = |C_{K-1}| =: m$ and $|C_K| = 1$, and we constrain that the output set takes at most one element from each class, which is a partition matroid constraint. For each class i, there is a unique "right" element, referred to as the red element of the class, and for the first $K-1$ classes, there are a large number of "wrong" elements, referred to as blue elements of the class. We will define a monotone submodular function $f\colon 2^E \to \mathbb{R}_+$ whose value is determined by (1) the presence/absence of the red element of class from 1 to K, and (2) the number of blue elements of class from 1 to $K-1$. More precisely, given a set S, we denote by r_i and b_i the numbers of red and blue elements of class i in S for $1 \le i \le K$, respectively, and then $f(S)$ takes the form of $f(r_1, r_2, \dots, r_K; b_1, b_2, \dots, b_K)$. We note that $r_i \in \{0, 1\}$ for all i, and b_K should always be 0. We call such a function *colorwise-symmetric with respect to the partition* $\{C_1, \dots, C_K\}$. We will assume that $f(1, \dots, 1; 0, \dots, 0)$ gives the optimal value.

In the input stream, for each $i \in \{1, \dots, K-1\}$ in this order, the blue and red elements of class i arrive in a random order, and then the unique red element of class K arrives. We design f so that it is hard to distinguish blue and red elements in each class. More precisely, f satisfies the property

$$\begin{aligned} &f(r_1, \dots, r_{i-1}, 1, 0, \dots, 0; b_1, \dots, b_{i-1}, b_i \quad , 0, \dots, 0) \\ =&f(r_1, \dots, r_{i-1}, 0, 0, \dots, 0; b_1, \dots, b_{i-1}, b_i + 1, 0, \dots, 0) \end{aligned}$$

for any $1 \le i \le K-1$, $r_1, \dots, r_{i-1} \in \{0, 1\}$, and $b_1, \dots, b_i \in \{0, 1, \dots, m\}$. Combined with the monotonicity of f, we can show that the maximum value we can obtain via any algorithm is $f(0, \dots, 0, 1; 1, \dots, 1, 0)$ with high probability. Again on the basis of some ideas suggested by computer simulations, we can construct f so that this value is small.

We note that we took a different approach from the information-theoretic argument based on communication complexity used to show existing lower bounds [35, 37], because we wanted to show lower bounds when the value oracle for a submodular function is available, and it is not clear how we can integrate it in the communication complexity setting. In [35], coverage functions were explicitly constructed from instances of a communication complexity problem, and hence we can regard that the sets used to define the coverage functions are given one by one in a streaming fashion, and we do not need the value oracle. In [37], the issue was avoided by assuming that the value oracle is weak.

Our algorithms Our algorithms for cardinality and matroid constraints, given in Theorems 1.4 and 1.5, all use branching, depending on the property of the first element o_1 of the optimal solution OPT in the stream. Here we explain the simplest case of cardinality constraint to highlight the basic ideas. We devise a general procedure which takes two parameters k and s. The former is the upper bound on the size of the optimal solution while the latter is the allowed size of the solution.

Such a procedure would guarantee that the returned solution achieves the approximation ratio of $\frac{s}{k+s-1}$, where we observe that the ratio improves when s is large relatively to k.

In the first branch, we assume that the value of o_1 is sufficiently large and we simply take the first element e whose value is above a certain threshold and then recurse on all the elements after e (with parameters $k-1$ and $s-1$). Doing this guarantees that the element we first take is of large value ("bang for the buck"), and more critically, o_1 (and hence the rest of OPT) is not "missed" in the recursion. In the second branch, we assume that the value of o_1 is too small and we can as well just focus on $\mathrm{OPT}-o_1$, by recursing directly on all remaining elements (with parameters $k-1$ and s). Even though the value of $\mathrm{OPT}-o_1$ is slightly smaller than OPT, the approximation ratio for the recursion is improved, as the available space s grows relatively to the optimal solution size.

For the case of matroid constraint, the above branching strategy need more careful handling. It is based on the idea of taking the first element that *resembles* o_1 and use it to recurse on $\mathrm{OPT}-o_1$. There is an extra issue that the element e resembling o_1 may not form an independent set together with $\mathrm{OPT}-o_1$. This issue is circumvented by using an extra set of candidates of o_1, based on a known fact in matroid theory.

1.3 Related work

Maximizing a monotone submodular function subject to various constraints is a subject that has been extensively studied in the literature. Although the problem is NP-hard even for a cardinality constraint, it can be approximated in polynomial time within a factor of $1-\frac{1}{e}$. See e.g., [3, 21, 22, 40]. On the other hand, even for a cardinality constraint, we need an exponential number of function evaluations to obtain approximation ratio better than $1-\frac{1}{e}$ [36, 39]. Also, even when the submodular function is explicitly given (as a coverage function), Feige [19] proved that the problem with a cardinality constraint cannot be approximated in polynomial time within a factor of $1-\frac{1}{e}+\varepsilon$ for any constant $\varepsilon > 0$ unless P is equal to NP. Besides a cardinality constraint, the problem has also been studied under (multiple) matroid constraint(s), p-system constraint, multiple knapsack constraints. See [9, 11, 12, 15, 16, 18, 20, 29, 32, 41] and the references therein.

Multi-pass streaming algorithms, where we are allowed to read a stream of the input multiple times, have also been studied [3, 10, 23, 25]. In particular, Chakrabarti and Kale [10] gave an $O(\varepsilon^{-3})$-pass streaming algorithms for a generalization of the maximum matching problem and the submodular maximization problem with cardinality constraint. Huang and Kakimura [23] designed an $O(\varepsilon^{-1})$-pass streaming algorithm with approximation guarantee $1/2-\varepsilon$ for the knapsack-constrained problem. Other than the streaming setting, recent applications of submodular function maximization to large data sets have motivated new directions of research on other computational models including parallel computation model such as the MapReduce model [7, 6, 30] and the adaptivity analysis [4, 5, 14, 17].

The maximum coverage problem is a special case of monotone submodular maximization under a cardinality constraint where the function is a set-covering function. For the special case, McGregor and Vu [35] and Batani et al. [8] gave a $(1-e^{-1}-\varepsilon)$-approximation algorithm in the multi-pass

streaming setting.

1.4 Organization

We prove our lower bound for strong-oracle algorithms for a cardinality constraint (Theorem 1.2) and a matroid constraint (Theorem 1.3) in Sections 2 and 3, respectively. We explain our weak-oracle algorithms and analyze them (Theorems 1.4 and 1.5) in Section 4.

2 Lower Bounds for Cardinality Constraints

In this section, we prove Theorem 1.2. As described in the introduction, the ground set E is partitioned into a blue set B, a red set R, and a purple set P, where $|B| = n-K$, $|R| = K-1$, and $|P| = 1$. We design a colorwise-symmetric function $f\colon 2^E \to \mathbb{Z}_+$ such that $f(b+1,0,0) = f(b,1,0)$ for any non-negative integer $b \le |B|-1$ and the values $f(K,0,0)$, $f(K-1,1,0)$ and $f(K-1,0,1)$ are small. More specifically, we show the following:

Lemma 2.1. *For any large enough integer n and any integer $h \ge K$, there exists a colorwise-symmetric function $f\colon 2^E \to \mathbb{Z}_+$ with $|E| = n$ that satisfies the following conditions.*

- *(i) f is monotone submodular.*
- *(ii) [Indistinguishability] $f(b+1,0,0) = f(b,1,0)$ holds for all $0 \le b \le n-K-1$.*
- *(iii) [Output value] $f(K,0,0) = f(K-1,1,0) = hK + \frac{(K-1)K}{2}$ and $f(K-1,0,1) = (K-1)^2 + \frac{h(h+1)}{2}$ hold.*
- *(iv) [Optimal value] $f(0,K-1,1) = (K-1)(h+K-1) + \frac{h(h+1)}{2}$ holds.*

We defer the construction of our hard function and its analysis to Sections 2.1–2.3.

Below we prove Theorem 1.2 using Lemma 2.1. We will use the following bound in the proof.

Proposition 2.2. *We have*

$$\left(1 - \frac{k^2}{n}\right)\frac{n^k}{k!} \le \binom{n}{k} \le \frac{n^k}{k!}.$$

Proof. The claim holds from

$$\binom{n}{k} = \frac{\prod_{i=0}^{k-1}(n-i)}{k!}$$

and

$$n^k \ge \prod_{i=0}^{k-1}(n-i) \ge (n-k)^k = \left(1-\frac{k}{n}\right)^k n^k \ge \left(1-\frac{k^2}{n}\right)n^k. \qquad \square$$

Proof of Theorem 1.2. Let $h \ge K$ be an integer determined later, and let $f\colon 2^E \to \mathbb{Z}_+$ with $|E| = n$ be the colorwise-symmetric function as in Lemma 2.1.

Let $\mathcal{D}$ be the uniform distribution over orderings $(e_1,\dots,e_n)$ of elements of E, conditioned on that $e_1,\dots,e_{n-1}$ include all the red and blue elements. Note that e_n is the (unique) purple

element. By Yao's minimax principle, to prove Theorem 1.2, it suffices to show that any deterministic streaming algorithm A with $o\left(\frac{n}{K^2}\right)$ space on an input sampled from $\mathcal{D}$ does not achieve approximation ratio more than $\frac{K}{2K-1}$ in expectation.

Let $(e_1, \ldots, e_n)$ denote a sequence of elements sampled from $\mathcal{D}$. Let S_t be the set of elements that A holds after the t-th step, that is, the t-th element e_t has arrived and A has updated the set of elements it holds (by adding e_t and/or discarding elements already in the set). We define $S_0 = \emptyset$ for convenience, and note that an algorithm chooses a subset of S_n as the output of A. Note also that, for each $1 \leq t \leq n$, the set S_t is completely determined by S_{t-1} and the values of $f(S)$ and $f(S \cup \{e_t\})$ $(S \subseteq S_{t-1})$, as A is deterministic.

For a set of indices $I \subseteq \{1, \ldots, n\}$, we define $S_I = \{e_i \mid i \in I\}$. Then, for $t \in \{0, 1, \ldots, n-1\}$, we iteratively define a *canonical set* I_t^* of indices (not elements) as follows. First, we set $I_0^* = \emptyset$. Then, for each $1 \leq t \leq n-1$, we define I_t^* as the set of indices of elements in S_t when A had $S_{I_{t-1}^*}$ after the $(t-1)$-th step and all but at most one element in $S_{I_{t-1}^*} \cup \{e_t\}$ are blue. Note that I_t^* is uniquely determined because A is deterministic, and by Property (ii) of Lemma 2.1, the value of $f(S_I \cup \{e_t\})$ for $I \subseteq I_{t-1}^*$ is uniquely determined from the size of I.

We say that A followed the *canonical process* if A holds the set $S_{I_t^*}$ after the t-th step for each $1 \leq t \leq n-1$. For $1 \leq t \leq n-1$, let X_t be the event that $S_{I_{t-1}^*}$ has one or more red elements and e_t is red. Then, the probability that A does not follow the canonical process is bounded by the probability that $\bigvee_{t=1}^{n-1} X_t$ happens. First, we have

$$\Pr[X_t] \leq \frac{\sum_{r=1}^{K-2} \binom{s}{r}\binom{n-s-2}{K-2-r}}{\binom{n-1}{K-1}},$$

where s is the space usage of the algorithm, because the probability that $S_{I_{t-1}^*}$ has r red balls and e_t is red is at most $\binom{s}{r}\binom{n-s-2}{K-2-r}/\binom{n-1}{K-1}$. Then by a union bound, we have

$$\begin{aligned}
\Pr\left[\bigvee_{t=1}^{n-1} X_t\right] &\leq \sum_{t=1}^{n-1} \Pr[X_t] \leq (n-1)\frac{\sum_{r=1}^{K-2} \binom{s}{r}\binom{n-s-2}{K-2-r}}{\binom{n-1}{K-1}} \\
&\leq (n-1)\frac{\sum_{r=1}^{K-2} \frac{s^r}{r!}\frac{(n-s-2)^{K-2-r}}{(K-2-r)!}}{\left(1-\frac{K^2}{n}\right)\frac{(n-1)^{K-1}}{(K-1)!}} && \text{(By Proposition 2.2)} \\
&= \frac{K-1}{\left(1-\frac{K^2}{n}\right)(n-1)^{K-2}} \sum_{r=1}^{K-2} \binom{K-2}{r} s^r (n-s-2)^{K-2-r} \\
&= \frac{K-1}{\left(1-\frac{K^2}{n}\right)(n-1)^{K-2}} \left((n-2)^{K-2} - (n-s-2)^{K-2}\right) \\
&= \frac{K-1}{1-\frac{K^2}{n}} \left(\frac{n-2}{n-1}\right)^{K-2} \left(1 - \left(1 - \frac{s}{n-2}\right)^{K-2}\right) \\
&\leq \frac{K-1}{1-\frac{K^2}{n}} \frac{s(K-2)}{n-2} && \text{(By } (1-x)^d \geq 1-dx\text{)} \\
&= \frac{1}{1-\frac{K^2}{n}} O\left(\frac{K^2 s}{n}\right).
\end{aligned}$$

Let Y be the event that $S_{I^*_{n-1}}$ has one or more red elements. We have

$$\begin{aligned}
\Pr[Y] &\le 1 - \frac{\binom{n-s-1}{K-1}}{\binom{n-1}{K-1}} \le 1 - \frac{\left(1-\frac{K^2}{n-s}\right)\frac{(n-s-1)^{K-1}}{(K-1)!}}{\frac{(n-1)^{K-1}}{(K-1)!}} && \text{(By Proposition 2.2)} \\
&= 1 - \left(1-\frac{K^2}{n-s}\right)\left(1-\frac{s}{n-1}\right)^{K-1} \\
&\le 1 - \left(1-\frac{K^2}{n-s}\right)\left(1-\frac{s(K-1)}{n-1}\right) && \text{(By } (1-x)^d \ge 1-dx\text{)} \\
&= \frac{K^2}{n-s} + \left(1-\frac{K^2}{n-s}\right)\frac{s(K-1)}{n-1}.
\end{aligned}$$

As long as $K = o(\sqrt{n})$ and $s = o\left(\frac{n}{K^2}\right)$, the probability that none of $X_1, \ldots, X_{n-1}$, and Y happens is at least $1 - o(1)$ by setting the hidden constant in s to be small enough. If none of the events has happened, the algorithm A can only obtain values for sets S with $S \subseteq S_{I^*_{t-1}} \cup \{e_t\}$ for some $1 \le t \le n$ and $|S| \le K$. As $S_{I^*_{t-1}} \cup \{e_t\}$ for any $1 \le t \le n-1$ contains at most one red element and $S_{I^*_{n-1}}$ contains no red element, the value of $f(S)$ is upper-bounded by $\max\{f(K,0,0), f(K-1,1,0), f(K-1,0,1)\}$, which is given by Property (iii) of Lemma 2.1. Recall that the optimal value is given by Property (iv) of Lemma 2.1. Therefore, the approximation ratio (in expectation over $\mathcal{D}$) is

$$(1-o(1)) \cdot \frac{\max\left\{hK + \frac{(K-1)K}{2}, (K-1)^2 + \frac{h(h+1)}{2}\right\}}{(K-1)(h+K-1) + \frac{h(h+1)}{2}} + o(1) \cdot 1.$$

The ratio is minimized when h is $\lfloor\sqrt{2}(K-1)\rfloor$ or $\lceil\sqrt{2}(K-1)\rceil$. When K approaches to infinity, the ratio is

$$(1-o(1)) \cdot \frac{2}{2+\sqrt{2}} + o(1) > 0.585,$$

as desired. □

2.1 Construction of Hard Functions

We first define our function, and then describe the intuition behind the construction. The next two subsections give its correctness proof.

Definition 2.3. *We define a colorwise-symmetric function $f\colon 2^E \to \mathbb{Z}_+$ recursively by its marginal return: Define*

$$f(0,0,0) = 0 \quad \textit{and} \quad f(0,0,1) = \frac{h(h+1)}{2}.$$

We denote the marginal returns of f by

$$\begin{aligned}
\Delta_{\mathrm{r}}(b,r) &= f(b,r+1,0) - f(b,r,0) = f(b,r+1,1) - f(b,r,1), \\
\Delta_{\mathrm{b}}(b,0,0) &= f(b+1,0,0) - f(b,0,0), \\
\Delta_{\mathrm{b}}(b,0,1) &= f(b+1,0,1) - f(b,0,1),
\end{aligned}$$

where they are defined by

$$\Delta_{\mathrm{r}}(b,r)=\begin{cases}K-1+h-b & \text{if} \quad 0\le b\le h+r,\\ K-1-\left\lceil\frac{r+b-h}{2}\right\rceil & \text{if} \quad h+r+1\le b\le h+2(K-2)-r,\\ 0 & \text{if} \quad h+2(K-2)-r+1\le b,\end{cases}$$

$$\Delta_{\mathrm{b}}(b,0,0)(=\Delta_{\mathrm{r}}(b,0))=\begin{cases}K-1+h-b & \text{if} \quad 0\le b\le h,\\ K-1-\left\lceil\frac{b-h}{2}\right\rceil & \text{if} \quad h+1\le b\le h+2(K-2),\\ 0 & \text{if} \quad h+2(K-2)+1\le b,\end{cases}$$

$$\Delta_{\mathrm{b}}(b,0,1)=\begin{cases}K-1 & \text{if} \quad 0\le b\le h,\\ K-1-\left\lceil\frac{b-h}{2}\right\rceil & \text{if} \quad h+1\le b\le h+2(K-2),\\ 0 & \text{if} \quad h+2(K-2)+1\le b.\end{cases}$$

The value $f(b,r,p)$ is determined in the following way: we start with the "base value", $f(0,0,p)$ (the presence of the purple element), then add the blue elements one by one until there are b of them (each increasing the marginal value by $\Delta_{\mathrm{b}}(i,0,p)$ for $0\le i\le b-1$), and then add the red elements one by one until there are r of them (each increasing the marginal value by $\Delta_{\mathrm{r}}(b,i)$ for $0\le i\le r-1$). In other words, we have

$$f(b,r,p)=f(0,0,p)+\sum_{j=0}^{b-1}\Delta_{\mathrm{b}}(j,0,p)+\sum_{i=0}^{r-1}\Delta_{\mathrm{r}}(b,i). \tag{3}$$

2.1.1 Ideas behind the function

We start with several simple observations. The marginal values $\Delta_{\mathrm{b}}(b,0,0)$, $\Delta_{\mathrm{b}}(b,0,1)$ and $\Delta_{\mathrm{r}}(b,r)$ are *monotonically non-increasing* as the parameters b and r increase[1]. This conforms with the property of diminishing marginal return of a submodular function. The next thing to notice is that, by design, the values $\Delta_{\mathrm{b}}(b,0,0)$ are exactly identical to $\Delta_{\mathrm{r}}(b,0)$. So $f(b+1,0,0)=f(b,0,0)+\Delta_{\mathrm{b}}(b,0,0)=f(b,0,0)+\Delta_{\mathrm{r}}(b,0)=f(b,1,0)$ and this is required by Lemma 2.1(ii).

We next explain why the marginal values $\Delta_{\mathrm{b}}(b,0,0)$, $\Delta_{\mathrm{b}}(b,0,1)$ and $\Delta_{\mathrm{r}}(b,r)$ are chosen in such a manner. A concrete example can be very useful to reveal the patterns generated by them. Assume that $K=4$ (thus 3 red elements and 1 purple element, and a large number of blue elements) and we choose $h=4$. Table 2 gives the function values when the purple element is absent or present, along with $\Delta_{\mathrm{r}}(b,r)$, the marginal value of adding a red element. We observe that for every fixed r, $\Delta_{\mathrm{r}}(b,r)$ is *strictly* monotonically decreasing in b until $b=h+r$ (we color it differently); when b is

[1]To see $\Delta_{\mathrm{r}}(b,r)\ge\Delta_{\mathrm{r}}(b+1,r)$, it suffices to show when $b=h+r$ and when $b=h+2(K-2)-r$. They follow because $\Delta_{\mathrm{r}}(h+r,r)=K-r-1\ge K-1-\lceil r+\frac{1}{2}\rceil=\Delta_{\mathrm{r}}(h+r+1,r)$ and $\Delta_{\mathrm{r}}(h+2(K-2)-r,r)=K-1-\lceil K-1\rceil\ge 0=\Delta_{\mathrm{r}}(h+2(K-2)-r+1,r)$. To see $\Delta_{\mathrm{r}}(b,r)\ge\Delta_{\mathrm{r}}(b,r+1)$, the only tricky cases are when $b=h+r+1$ and when $b=h+2(K-2)-r$. It holds that $\Delta_{\mathrm{r}}(h+r+1,r)=K-1-r-1=\Delta_{\mathrm{r}}(h+r+1,r+1)$ and $\Delta_{\mathrm{r}}(h+2(K-2)-r,r)=1\ge 0=\Delta_{\mathrm{r}}(h+2(K-2)-r,r+1)$.

beyond $h+r$, the same value $\Delta_{\mathrm{r}}(b,r)$ appears twice before decreasing. Notice that the smaller the $r=i$, $\Delta_{\mathrm{r}}(b,i)$ decreases in b more slowly (conforming with the submodularity).

Table 2: The function values along with the marginal value $\Delta_{\mathrm{r}}(b,r)$ when $K=h=4$. After the value $b=h+r$ (we color it specially), the same $\Delta_{\mathrm{r}}(b,r)$'s appear twice before decreasing. Also notice that the values $f(b+1,0,0)=f(b,1,0)$ for all $0\le b\le n-K-1$ (Lemma 2.1(ii)).

$p=0$	$r=0$		$r=1$		$r=2$		$r=3$
b	f	Δ_{r}	f	Δ_{r}	f	Δ_{r}	f
0	0	7	7	7	14	7	21
1	7	6	13	6	19	6	25
2	13	5	18	5	23	5	28
3	18	4	22	4	26	4	30
4	22	3	25	3	28	3	31
5	25	2	27	2	29	2	31
6	27	2	29	1	30	1	31
7	29	1	30	1	31	0	31
8	30	1	31	0	31	0	31
9	31	0	31	0	31	0	31
10	31	—	31	—	31	—	31

$p=1$	$r=0$		$r=1$		$r=2$		$r=3$
b	f	Δ_{r}	f	Δ_{r}	f	Δ_{r}	f
0	13	7	20	7	27	7	34
1	16	6	22	6	28	6	34
2	19	5	24	5	29	5	34
3	22	4	26	4	30	4	34
4	25	3	28	3	31	3	34
5	28	2	30	2	32	2	34
6	30	2	32	1	33	1	34
7	32	1	33	1	34	0	34
8	33	1	34	0	34	0	34
9	34	0	34	0	34	0	34
10	34	—	34	—	34	—	34

Indeed we choose all marginal values $\Delta_{\mathrm{r}}(b,r)$, $\Delta_{\mathrm{b}}(b,0,0)$ and $\Delta_{\mathrm{b}}(b,0,1)$ with two objectives. (1) The indistinguishability of red and blue elements (i.e., Lemma 2.1(ii)), which forces $\Delta_{\mathrm{r}}(b,0)=\Delta_{\mathrm{b}}(b,0,0)$, and (2) creating a particular pattern of $f(b,K-1,0)$ and $f(b,K-1,1)$. As can be observed in this example, $f(b,K-1,1)$ is simply the optimal value for all b, while $f(b,K-1,0)$ increases by the amount of h, $h-1$, down to 1 when b increases from 0 to h and stop growing when b is beyond h. The fact that we want $f(b,K-1,0)$ to stop growing when b is beyond h explains why $\Delta_{\mathrm{r}}(b,r)$ behaves somehow differently when b is beyond $h+r$.

2.2 Correctness of Function Values

In this section, we show that the function f defined in Definition 2.3 satisfies Lemma 2.1 (ii)–(iv).

Since $\Delta_{\mathrm{b}}(b,0,0)=\Delta_{\mathrm{r}}(b,0)$ for each b, we immediately have (ii) of Lemma 2.1. Moreover, it follows from (3) that

$$f(K,0,0)=\sum_{j=0}^{K-1}\Delta_{\mathrm{b}}(j,0,0)=hK+\frac{(K-1)K}{2},$$

$$f(K-1,1,0)=\Delta_{\mathrm{r}}(K-1,0)+\sum_{j=0}^{K-2}\Delta_{\mathrm{b}}(j,0,0)=\sum_{j=0}^{K-1}\Delta_{\mathrm{b}}(j,0,0)=hK+\frac{(K-1)K}{2},$$

$$f(K-1,0,1) = \sum_{j=0}^{K-2} \Delta_{\mathrm{b}}(j,0,1) + f(0,0,1) = (K-1)(K-1) + \frac{h(h+1)}{2}.$$

It also holds that

$$f(0,K-1,1) = \sum_{i=0}^{K-2} \Delta_{\mathrm{r}}(0,i) + f(0,0,1) = (K-1)(h+K-1) + \frac{h(h+1)}{2}.$$

Thus (iii) and (iv) of Lemma 2.1 follow.

2.3 Monotonicity and Submodularity

Below we prove Lemma 2.1 (i), that is, the function f defined in Definition 2.3 is monotone and submodular. To this end, it suffices to show the following lemma.

Lemma 2.4. *Let $f\colon 2^E \to \mathbb{Z}_+$ be the colorwise-symmetric function defined in Definition 2.3. Then we have*

1. $f(b_1,r_1,1) - f(b_1,r_1,0) \geq f(b_2,r_2,1) - f(b_2,r_2,0) \geq 0$ *for* $0 \leq b_1 \leq b_2 \leq |B|$ *and* $0 \leq r_1 \leq r_2 \leq |R|$.
2. $f(b_1,r_1+1,p_1) - f(b_1,r_1,p_1) \geq f(b_2,r_2+1,p_2) - f(b_2,r_2,p_2) \geq 0$ *for* $0 \leq b_1 \leq b_2 \leq |B|$, $0 \leq r_1 \leq r_2 \leq |R|-1$, *and* $0 \leq p_1 \leq p_2 \leq 1$.
3. $f(b_1+1,r_1,p_1) - f(b_1,r_1,p_1) \geq f(b_2+1,r_2,p_2) - f(b_2,r_2,p_2) \geq 0$ *for* $0 \leq b_1 \leq b_2 \leq |B|-1$, $0 \leq r_1 \leq r_2 \leq |R|$, *and* $0 \leq p_1 \leq p_2 \leq 1$.

Proof. **(1)** By definition, it holds that, for $\ell = 1, 2$,

$$f(b_\ell,r_\ell,1) - f(b_\ell,r_\ell,0) = \sum_{j=0}^{b_\ell-1} \Big(\Delta_{\mathrm{b}}(j,0,1) - \Delta_{\mathrm{b}}(j,0,0)\Big) + f(0,0,1) - f(0,0,0).$$

Since $\Delta_{\mathrm{b}}(j,0,0) = \Delta_{\mathrm{b}}(j,0,1)$ if $j \geq h+1$, the RHS is equal to

$$\sum_{j=0}^{\min\{b_\ell-1,h\}} \Big(K-1-(K-1+h-j)\Big) + \frac{h(h+1)}{2} \geq -\sum_{j=0}^{h}(h-j) + \frac{h(h+1)}{2} = 0.$$

Hence the marginal return with respect to p is non-negative. Moreover,

$$\Big(f(b_1,r_1,1) - f(b_1,r_1,0)\Big) - \Big(f(b_2,r_2,1) - f(b_2,r_2,0)\Big) = \sum_{j=\min\{h,b_1-1\}}^{\min\{h,b_2-1\}} (h-j) \geq 0,$$

since $h-j \geq 0$ for $j \leq h$. Thus (1) holds.

(2) We observe that $\Delta_{\mathrm{r}}(b,r)$ is a monotonically non-increasing function with respect to b, and a monotonically non-increasing function with respect to r (See the footnote in Section 2.1.1). Hence it holds that

$$\Delta_{\mathrm{r}}(b_1,r_1) \geq \Delta_{\mathrm{r}}(b_1,r_2) \geq \Delta_{\mathrm{r}}(b_2,r_2) \geq 0.$$

This implies that

$$f(b_1, r_1 + 1, p_1) - f(b_1, r_1, p_1) = \Delta_{\mathrm{r}}(b_1, r_1) \geq \Delta_{\mathrm{r}}(b_2, r_2) = f(b_2, r_2 + 1, p_2) - f(b_2, r_2, p_2) \geq 0.$$

Thus (2) holds.

(3) We first observe the following claims.

Claim 1. *Let* $0 \leq r \leq |R|$.

1. *For* $0 \leq b \leq h + r - 1$, *it holds that* $2\Delta_{\mathrm{r}}(b + 1, r) = \Delta_{\mathrm{r}}(b, r) + \Delta_{\mathrm{r}}(b + 2, r)$.
2. *For* $h + r \leq b$, *it holds that* $\Delta_{\mathrm{r}}(b, r) = \Delta_{\mathrm{r}}(b + 1, r - 1)$.

Proof. They follow from the definition of Δ_{r}. (1) clearly holds when $b + 2 \leq h + r$, because Δ_{r} is a linear function with respect to b . When $b = h + r - 1$, (1) also holds since $2\Delta_{\mathrm{r}}(b + 1, r) = 2(K - 1 - r)$, $\Delta_{\mathrm{r}}(b, r) = K - r$, and $\Delta_{\mathrm{r}}(b + 2, r) = K - 1 - (r + 1)$. For (2), the equality holds when $h + r + 1 \leq b$, since Δ_{r} depends on $b + r$. Moreover, when $h + r = b$, the equality holds since $\Delta_{\mathrm{r}}(b, r) = \Delta_{\mathrm{r}}(b + 1, r - 1) = K - 1 - r$. □

Claim 2. *For* $0 \leq b \leq |B| - 2$, $0 \leq r \leq |R|$, *and* $0 \leq p \leq 1$, *it holds that*

$$f(b + 1, r, p) - f(b, r, p) \geq f(b + 2, r, p) - f(b + 1, r, p) \geq 0.$$

Proof. We first consider the case when $b \leq h + r - 2$. We will show that the first inequality holds by induction on r. The inequality holds when $r = 0$ as $\Delta_{\mathrm{b}}(b, 0, p) \geq \Delta_{\mathrm{b}}(b + 1, 0, p)$. Suppose that $r > 0$. Then we have

$$\begin{aligned} f(b + 1, r, p) - f(b, r, p) &= \Big(f(b + 1, r - 1, p) + \Delta_{\mathrm{r}}(b + 1, r - 1)\Big) - \Big(f(b, r - 1, p) + \Delta_{\mathrm{r}}(b, r - 1)\Big) \\ &\geq f(b + 2, r - 1, p) + \Delta_{\mathrm{r}}(b + 1, r - 1) - f(b + 1, r - 1, p) - \Delta_{\mathrm{r}}(b, r - 1) \\ &= f(b + 2, r, p) - f(b + 1, r, p) \\ &\quad + 2\Delta_{\mathrm{r}}(b + 1, r - 1) - \Delta_{\mathrm{r}}(b, r - 1) - \Delta_{\mathrm{r}}(b + 2, r - 1), \end{aligned}$$

where the inequality holds by the induction hypothesis. Therefore, it follows from Claim 1 (1) that for $0 \leq b \leq h + r - 2$,

$$f(b + 1, r, p) - f(b, r, p) \geq f(b + 2, r, p) - f(b + 1, r, p).$$

Thus, the claimed inequality holds when $b \leq h + r - 2$. This implies that, for $0 \leq b \leq h + r - 2$, we have $f(b + 1, r, p) - f(b, r, p) \geq f(h + r + 1, r, p) - f(h + r, r, p)$.

Next suppose that $h + r - 1 \leq b$. We may assume that $r > 0$, as the case when $r = 0$ easily follows. By definition, it holds that

$$f(b + 1, r, p) - f(b, r, p) = \sum_{i=0}^{r-1} \Big(\Delta_{\mathrm{r}}(b + 1, i) - \Delta_{\mathrm{r}}(b, i)\Big) + \Delta_{\mathrm{b}}(b, 0, p).$$

Since $\Delta_{\mathrm{r}}(b,i) = \Delta_{\mathrm{r}}(b+1,i-1)$ for $i = 1,\dots,r-1$ by Claim 1 (2), we have

$$f(b+1,r,p) - f(b,r,p) = \Delta_{\mathrm{r}}(b+1,r-1) - \Delta_{\mathrm{r}}(b,0) + \Delta_{\mathrm{b}}(b,0,p) = \Delta_{\mathrm{r}}(b+1,r-1),$$

where the last equality follows from $\Delta_{\mathrm{r}}(b,0) = \Delta_{\mathrm{b}}(b,0,p)$ for $b \geq h$. Similarly, $f(b+2,r,p) - f(b+1,r,p) = \Delta_{\mathrm{r}}(b+2,r-1)$. Hence

$$f(b+1,r,p) - f(b,r,p) = \Delta_{\mathrm{r}}(b+1,r-1) \geq \Delta_{\mathrm{r}}(b+2,r-1) = f(b+2,r,p) - f(b+1,r,p).$$

Thus the claimed inequality holds.

Moreover, for $h+r-1 \leq b \leq |B|-2$, we have $f(b+1,r,p) - f(b,r,p) \geq f(|B|,r,p) - f(|B|-1,r,p) = \Delta_{\mathrm{r}}(|B|,r-1) \geq 0$. Thus the monotonicity also holds. □

Claim 3. *For $0 \leq b \leq |B|-1$, $0 \leq r \leq |R|$, and $0 \leq p \leq 1$, it holds that*

$$f(b+1,r,0) - f(b,r,0) \geq f(b+1,r,1) - f(b,r,1).$$

Proof. We will show the claim by induction on r. The inequality holds when $r = 0$ as $\Delta_{\mathrm{b}}(b,0,0) \geq \Delta_{\mathrm{b}}(b,0,1)$. Suppose that $r > 0$. Then

$$\begin{aligned} f(b+1,r,0) - f(b,r,0) &= (f(b+1,r-1,0) + \Delta_{\mathrm{r}}(b+1,r-1)) - (f(b,r-1,0) + \Delta_{\mathrm{r}}(b,r-1)) \\ &\geq f(b+1,r-1,1) + \Delta_{\mathrm{r}}(b+1,r-1) - f(b,r-1,1) - \Delta_{\mathrm{r}}(b,r-1) \\ &= f(b+1,r,1) - f(b,r,1), \end{aligned}$$

where the inequality holds by the induction hypothesis. Thus the claim holds. □

It holds that, for $0 \leq b \leq |B|-1$, $0 \leq r_1 \leq r_2 \leq |R|$, and $0 \leq p \leq 1$,

$$f(b+1,r_1,p) - f(b,r_1,p) \geq f(b+1,r_2,p) - f(b,r_2,p), \tag{4}$$

since $f(b+1,r_\ell,p) - f(b,r_\ell,p) = \sum_{i=0}^{r_\ell-1} (\Delta_{\mathrm{r}}(b+1,i) - \Delta_{\mathrm{r}}(b,i)) + \Delta_{\mathrm{b}}(b,0,p)$ for $\ell = 1,2$.

Therefore, applying Claims 2 and 3 with (4), we have

$$\begin{aligned} f(b_1+1,r_1,p_1) - f(b_1,r_1,p_1) &\geq f(b_1+1,r_2,p_1) - f(b_1,r_2,p_1) \\ &\geq f(b_2+1,r_2,p_1) - f(b_2,r_2,p_1) \geq f(b_2+1,r_2,p_2) - f(b_2,r_2,p_2). \end{aligned}$$

Since the monotonicity follows by Claim 2, we complete the proof of (3). □

3 Lower Bounds for Matroid Constraints

In this section, we prove Theorem 1.3. As described in the introduction, we assume that the ground set E is partitioned into classes $C_1,\dots,C_K$ such that $|C_1| = \cdots = |C_{K-1}| =: m$ and $|C_K| = 1$. Note that the number of elements $n := |E|$ is $(K-1)m+1$. We design a monotone submodular function $f\colon 2^E \to \mathbb{Z}_+$ that is colorwise-symmetric with respect to the partition $\{C_1,\dots,C_K\}$ such that it is hard to distinguish blue and red elements in each class whereas we need to hit all the red elements to get the optimal value. We will specify the exact values of f in the next section. Here, we summarize the critical properties of f and use them to prove Theorem 1.3.

Lemma 3.1. *For any large enough integer m and any positive integer K, there exists a colorwise-symmetric function $f\colon 2^E \to \mathbb{Z}_+$ with respect to a partition $E = \bigcup_{i=1}^{K} C_i$ with $|C_1| = \cdots = |C_{K-1}| = m$ and $|C_K| = 1$ such that*

(i) f is monotone submodular.

(ii) [Optimal value] $f(1,\dots,1;0,\dots,0) = (2K-1)!$.

(iii) [Output value] $f(0,\dots,0,1;1,\dots,1,0) = K(2K-2)!$.

(iv) [Indistinguishability] For any $1 \le i \le K-1$ and $r_1,\dots,r_{i-1},b_1,\dots,b_i$, we have

$$\begin{aligned} &f(r_1,\dots,r_{i-1},1,0,\dots,0;b_1,\dots,b_{i-1},b_i \quad ,0,\dots,0) \\ =&f(r_1,\dots,r_{i-1},0,0,\dots,0;b_1,\dots,b_{i-1},b_i+1,0,\dots,0). \end{aligned}$$

Proof of Theorem 1.3. We modify the proof of Theorem 1.2.

Let $f\colon 2^E \to \mathbb{Z}_+$ be the function as in Lemma 3.1. For $1 \le i \le K-1$, let $T_i = \{(i-1)m+1,\dots,im\}$ and let $T_K = \{n\}$. Let $\mathcal{D}$ be the uniform distribution over orderings $(e_1,\dots,e_n)$ of elements of E, conditioned on that for each $1 \le i \le K$, the set $\{e_t \mid t \in T_i\}$ consists of all the elements of class i. By Yao's minimax principle, to prove Theorem 1.3, it suffices to show that any deterministic streaming algorithm A with $o\left(\frac{n}{K}\right)$ space on an input sampled from $\mathcal{D}$ does not achieve approximation ratio more than $\frac{K}{2K-1}$ in expectation.

We define S_t for $t \in \{0,1,\dots,n\}$ and S_I for $I \subseteq \{1,\dots,n\}$ as in the proof of Theorem 1.2. For $t \in \{0,1,\dots,n-1\}$, we iteratively define a *canonical set* I_t^* of indices (not elements) as follows. First, we set $I_0^* = \emptyset$. Then, for each $1 \le t \le n-1$, we define I_t^* as the set of indices of elements in S_t when A had $S_{I_{t-1}^*}$ after the $(t-1)$-th step and all but at most one element in $S_{I^*} \cup \{e_t\}$ are blue. Note that I_t^* is uniquely determined because A is deterministic, and by Property (iv) of Lemma 3.1, the value of $f(S_I \cup \{e_t\})$ for $I \subseteq I_t^*$ is uniquely determined from the sizes of $I \cap T_1,\dots,I \cap T_{K-1}$.

We say that A followed the *canonical process* if A holds the set $S_{I_t^*}$ after the t-th step for each $1 \le t \le n-1$. For $1 \le t \le n-1$, let X_t be the event that $S_{I_{t-1}^*}$ has one or more red elements and e_t is red. Then, the probability that A does not follow the canonical process is bounded by the probability that $\bigvee_{t=1}^{n-1} X_t$ happens.

To bound $\Pr[X_t]$, we introduce some notations. Let s be the space usage of the algorithm. For $i \in \{1,\dots,K-1\}$, let $s_i = |I_{t-1}^* \cap T_i|$. For $t \in \{1,\dots,n-1\}$, let $i_t \in \{1,\dots,K-1\}$ be the class that the t-th element belongs to, that is, the unique integer i with $t \in T_i$. Then for any $t \in \{1,\dots,n-1\}$, we have

$$\begin{aligned} \Pr[X_t] &\le \sum_{r=1}^{K-2} \max_{\substack{r_1,\dots,r_{K-1} \in \{0,1\}: \\ \sum_{i \ne i_t} r_i \le r}} \frac{s_{i_t}}{m} \prod_{i \ne i_t} \frac{s_i^{r_i}(m-s_i)^{1-r_i}}{m} \le \sum_{r=1}^{K-2} \left(\frac{s}{r}\right)^r \frac{m^{K-2-r}}{m^{K-1}} \\ &= \frac{1}{m} \sum_{r=1}^{K-2} \left(\frac{s}{rm}\right)^r \le \frac{s}{m(m-s)} = O\left(\frac{s}{m^2}\right). \end{aligned}$$

Here, we regard r as the number of red elements in $S_{I_{t-1}^*}$ and regard $r_1,\dots,r_{K-1} \in \{0,1\}$ as the numbers of red elements in $S_{I_{t-1}^* \cap T_1},\dots,S_{I_{t-1}^* \cap T_{K-1}}$, respectively. The first inequality holds

because the probability that e_t is red is $\frac{s_{i_t}}{m}$ and the probability that $S_{I^*_{t-1}\cap T_i}$ has r_i red elements is $\frac{s_i^{r_i}(m-s_i)^{1-r_i}}{m}$.

Now by a union bound, we have

$$\Pr\left[\bigvee_{t=1}^{n-1} X_t\right] \leq \sum_{t=1}^{n-1} \Pr[X_t] = O\left(\frac{sn}{m^2}\right) = O\left(\frac{Ks}{m}\right).$$

Let Y be the event that $S_{I^*_{n-1}}$ has one or more red elements. Then, we have

$$\Pr[Y] \leq 1 - \prod_{i=1}^{K-1} \frac{m-s_i}{m} \leq 1 - \frac{(m-s)m^{K-2}}{m^{K-1}} = \frac{s}{m}.$$

As long as $Ks = o(n)$, the probability that none of $X_1, \ldots, X_{n-1}$, and Y happens is at least $1-o(1)$ by setting the hidden constant in s to be small enough. If none of the events has happened, the algorithm A can only obtain values for sets S with $S \subseteq S_{I^*_{t-1}} \cup \{e_t\}$ for $1 \leq t \leq n$ and $|S| \leq K$. As $S_{I^*_{t-1}} \cup \{e_t\}$ for any $1 \leq t \leq n-1$ contains at most one red element and $S_{I^*_{n-1}}$ contains no red element, the values the algorithm can observe is at most $K(2K-2)!$ by Properties (iii) and (iv) of Lemma 3.1. Recall that the optimal value is $(2K-1)!$ by Property (ii) of Lemma 3.1. Therefore, the approximation ratio (in expectation over $\mathcal{D}$) is

$$(1-o(1)) \cdot \frac{K(2K-2)!}{(2K-1)!} + o(1) \cdot 1 = (1-o(1))\frac{K}{2K-1} + o(1). \qquad \square$$

3.1 Construction of the Hard Function

We begin by describing some characteristics of the function we will define. Let $\hat{b}_i = \min\{b_i, 2(K-i)\}$. The function will be expressed as a polynomial of r_i and $\hat{b}_i$ for all $1 \leq i \leq K$. In other words, the number of blue elements matter only up to a certain ceiling: For class i, if there are more than $2(K-i)$ blue elements in the class i, the function value is the same as if there are exactly $2(K-i)$ blue elements. To be more precise, we decree that

$$f(r_1, \cdots r_K; b_1, \ldots, b_K) = f(r_1, \ldots, r_K; \hat{b}_1, \ldots, \hat{b}_K).$$

We now define the function f recursively. For $t = 1, \ldots K$, let f_t be a function on the last t classes $C_{K-(t-1)}, \ldots, C_K$, that is, f_t takes the form of

$$f_t(r_{K-(t-1)}, \ldots, r_K; b_{K-(t-1)}, \ldots, b_K)$$

Define

$$f_1(r_K; b_K) = r_K,$$

and assume the function f_{t-1} is already defined for some $t \geq 2$. We then define the function f_t. For that purpose, we give some notation. Let $m_t = (2t-1)!$. As we will see later, the function

value of f_{t-1} is between 0 and m_{t-1}. Suppose that we are given $r_{K-(t-1)}, r_{K-(t-2)}, \ldots, r_K$ and $b_{K-(t-1)}, b_{K-(t-2)}, \ldots, b_K$. Define[2]

$$\delta_{t-1} = m_{t-1} - f_{t-1}(r_{K-(t-2)}, \ldots, r_K; \hat{b}_{K-(t-2)}, \ldots, \hat{b}_K).$$

We also define

$$d_t = 2(t-1) - \hat{b}_{K-(t-1)} \quad \text{and} \quad s_t = 1 - r_{K-(t-1)}.$$

The term d_t (resp., s_t) is simply the gap between $\hat{b}_{K-(t-1)}$ (resp., $r_{K-(t-1)}$) and its potential maximum. We remark that both of them are non-negative, and $\hat{b}_{K-(t-1)} + d_t = 2(t-1)$ and $r_{K-(t-1)} + s_t = 1$. We can then express f_t as follows:

$$f_t(r_{K-(t-1)}, \ldots, r_K; b_{K-(t-1)}, \ldots, b_K) = m_t - a_t \cdot d_t, \tag{5}$$

where

$$a_t = 2m_{t-1}s_t + \delta_{t-1}(d_t - 1).$$

The function f_t is set up in such a way so that $f_t(1, \ldots, 1; 0, \ldots, 0)$ is exactly $m_t = (2t-1)!$ for any t. As we will show, this maximizes f_t. On the other hand, $f_t(0, \ldots, 0; 0, \ldots, 0) = 0$, which is the minimum of f_t. To see the significance of the term a_t, recall that the term δ_{t-1} encodes the difference between the maximum of f_{t-1} and the actual value attained by the given $r_{K-(t-2)}, \ldots, r_K$ and $\hat{b}_{K-(t-2)}, \ldots, \hat{b}_K$ — therefore always non-negative. Then we can regard a_t as a linear combination of s_t and d_t (both are decided by the number of red/blue elements of class $K-(t-1)$), where the coefficients are respectively $2m_{t-1}$ and δ_{t-1} (both are decided by the number of red/blue elements from later classes $K-(t-1)+1, \ldots, K$). See Lemma 3.4 for a more precise summary of the above discussion.

3.2 Concrete Example and Some Observations

We present a concrete example to highlight several interesting properties of the function constructed in Section 3.1, and to share some of our experiences in searching for such a function. Let $K = 3$. All the function values when $0 \le b_1 \le 2(K-1) = 4$ and $0 \le b_2 \le 2(K-2) = 2$ are shown in Tables 3 and 4. We remark that having more blue elements does not increase the value further, as mentioned in Section 3.1.

We can first observe that, individually, the values of a single red/blue element of the first $K-1$ classes are all equal, and the value of the unique red element in the last class is half of them. In the present example, an element in the first two classes has value 48 while an element in the last class has value 24. In fact, we have the following observation for the constructed function f. Note that b_K should always be 0.

[2]Here we note that the terms δ_{t-1}, d_t, and s_t indeed depend on the value of $r_{K-(t-1)}, r_{K-(t-2)}, \ldots, r_K$ and $b_{K-(t-1)}, b_{K-(t-2)}, \ldots, b_K$. However, we choose to avoid the cumbersome notation of associating the former with the latter.

	$r_1=0, r_2=0$				$r_1=1, r_2=0$				$r_1=0, r_2=1$				$r_1=1, r_2=1$		
$\hat{b}_1$ \ $\hat{b}_2$	0	1	2	$\hat{b}_1$ \ $\hat{b}_2$	0	1	2	$\hat{b}_1$ \ $\hat{b}_2$	0	1	2	$\hat{b}_1$ \ $\hat{b}_2$	0	1	2
0	0	48	72	0	48	96	120	0	48	72	72	0	96	120	120
1	48	72	84	1	84	108	120	1	72	84	84	1	108	120	120
2	84	92	96	2	108	116	120	2	92	96	96	2	116	120	120
3	108	108	108	3	120	120	120	3	108	108	108	3	120	120	120
4	120	120	120	4	120	120	120	4	120	120	120	4	120	120	120

Table 3: When the unique element of C_3 is absent, i.e., $r_3 = 0$.

	$r_1=0, r_2=0$				$r_1=1, r_2=0$				$r_1=0, r_2=1$				$r_1=1, r_2=1$		
$\hat{b}_1$ \ $\hat{b}_2$	0	1	2	$\hat{b}_1$ \ $\hat{b}_2$	0	1	2	$\hat{b}_1$ \ $\hat{b}_2$	0	1	2	$\hat{b}_1$ \ $\hat{b}_2$	0	1	2
0	24	48	72	0	72	96	120	0	72	72	72	0	120	120	120
1	60	72	84	1	96	108	120	1	84	84	84	1	120	120	120
2	88	92	96	2	112	116	120	2	96	96	96	2	120	120	120
3	108	108	108	3	120	120	120	3	108	108	108	3	120	120	120
4	120	120	120	4	120	120	120	4	120	120	120	4	120	120	120

Table 4: When the unique element of C_3 is present, i.e., $r_3 = 1$.

Lemma 3.2. *The constructed function f satisfies the following.*

$$f(0,\dots,0,1,0,\dots,0;0,\dots,0) = \begin{cases} (2K-2)! & \text{if 1 is at the last class } K \\ 2(2K-2)! & \text{otherwise} \end{cases}$$

$$f(0,\dots,0;0,\dots,0,1,0,\dots,0) = 2(2K-2)!,$$

The proof will be given in Section 3.4.

In the present example, the optimal value is 120, which is reached by a set of $2(K-1)$ blue elements of class 1, i.e, $f(0,0,0;4,0,0) = 120$, though it is infeasible. The optimal value is also obtained by a set of all the red elements, which is equal to the sum of the values of each red element. In the example, we see that $f(1,1,1;0,0,0) = f(1,0,0;0,0,0)+f(0,1,0;0,0,0)+f(0,0,1;0,0,0) = 48+48+24 = 120$.

Lemma 3.3. *The constructed function f satisfies that*

$$f(1,\dots,1;0,\dots,0) = f(1,0,\dots,0;0,\dots,0) + f(0,1,0,\dots,0;0,\dots,0) + \cdots$$
$$\cdots + f(0,\dots,0,1;0,\dots,0) = (2K-1)!,$$
$$f(0,\dots,0;2(K-1),0,\dots,0) = (2K-1)!.$$

The proof will be given in Section 3.4.

By the construction, the function $f(r_1,r_2,r_3;b_1,b_2,b_3)$ can be expressed as the following polynomial:

$$f(r_1,r_2,r_3;b_1,b_2,b_3) = m_3 - (2m_2s_3 + (2m_1s_2 + s_1(d_2-1))\,d_2(d_3-1)) \cdot d_3$$

$$= 120 - (12s_3 + (2s_2 + s_1(d_2 - 1))\, d_2(d_3 - 1)) \cdot d_3, \tag{6}$$

where we recall that $m_t = (2t-1)!$, $d_t = 2(t-1) - \hat{b}_{K-(t-1)}$ and $s_t = 1 - r_{K-(t-1)}$ for $t = 1, 2, 3$. We can observe that f is a polynomial in s_i's and d_i's, where, for each monomial, s_i's have degree at most 1 and d_i's have degree at most 2. This implies that *the discrete second derivative with respect to b_i's is constant when b_i is in* $[0, 2(K-i)]$. Take the current example with $r_1 = r_2 = r_3 = b_2 = 0$. When b_1 increases from 0 to 4, the discrete first derivative is 48, 36, 24, 12 and the discrete second derivative is a constant, which is 12. It can be verified that the same property holds for all columns and all rows in Tables 3 and 4. In fact, we noticed in computer-aided search that imposing the additional constraint of constant second derivative does not change the ratio between the optimal value and the value an algorithm finds, making the found function more structured and generalizable. Based on computer-aided search imposing this additional constraint for small values of rank K, we found the recurrence in the previous section.

3.3 Correctness of the Function

We first show the monotonicity of the constructed function f_t, which implies Lemma 3.1 (i) and (ii) when $t = K$.

Lemma 3.4. *For any $t = 1, 2, \ldots, K$, the constructed function f_t satisfies the following.*

1. *f_t is monotone in $r_{K-(t-1)}, \ldots, r_K$ and in $b_{K-(t-1)}, \ldots, b_K$.*
2. *f_t reaches the maximum, which is $m_t = (2t-1)!$, when $r_{K-(t-1)} = \cdots = r_K = 1$.*
3. *f_t reaches the minimum, which is 0, when $r_{K-(t-1)} = \cdots = r_K = b_{K-(t-1)} = \cdots = b_K = 0$.*

Proof. We prove by induction on t. When $t = 1$, it is straightforward to verify the lemma. Consider when $t \geq 2$.

For (1), suppose that $r^1_{K-(t-1)} \geq r^2_{K-(t-1)}$, $\cdots$, $r^1_K \geq r^2_K$ and $b^1_{K-(t-1)} \geq b^2_{K-(t-1)}$, $\cdots$, $b^1_K \geq b^2_K$. For $\ell = 1, 2$, let δ^ℓ_{t-1}, d^ℓ_t, s^ℓ_t and f^ℓ_t denote the realizations of δ_{t-1}, d_t, s_t and f_t, based on $\bigcup_{j=K-(t-1)}^{K} \{r^\ell_j, b^\ell_j\}$. By the induction hypothesis, f_{t-1} is monotone, and hence we see $0 \leq \delta^1_{t-1} \leq \delta^2_{t-1}$. Furthermore, as $b^1_{K-(t-1)} \geq b^2_{K-(t-1)}$, we have $\hat{b}^1_{K-(t-1)} \geq \hat{b}^2_{K-(t-1)}$, implying that $d^1_t \leq d^2_t$. Now it holds by (5) that

$$\begin{aligned} f^1_t - f^2_t &= \delta^2_{t-1}(d^2_t - 1)d^2_t - \delta^1_{t-1}(d^1_t - 1)d^1_t + 2m_{t-1}\left(s^2_t d^2_t - s^1_t d^1_t\right) \\ &\geq \delta^2_{t-1}(d^2_t - 1)d^2_t - \delta^1_{t-1}(d^1_t - 1)d^1_t \\ &\geq \delta^1_{t-1}\left((d^2_t - 1)d^2_t - (d^1_t - 1)d^1_t\right) \geq 0 \end{aligned}$$

where the first inequality holds because $s^2_t \geq s^1_t$ and $d^2_t \geq d^1_t$, the second inequality is from $\delta^2_{t-1} \geq \delta^1_{t-1}$, and the last inequality follows from the fact that the function $x(x-1)$ is monotonically increasing for a non-negative integer x. (1) is then proved.

For (2), when $r_{K-(t-1)} = r_{K-(t-2)} = \cdots = r_K = 1$, by the induction hypothesis, f_{t-1} takes the maximum m_{t-1}, and hence $\delta_{t-1} = 0$. Then $a_t = 0$, since $s_t = 0$ and $\delta_{t-1} = 0$. Hence the f_t-value

in this case is equal to m_t. In addition, it holds that m_t is an upped bound of f_t, since $a_t d_t$ is non-negative in (5). Indeed, $a_t d_t = 0$ when $d_t = 0$, and when $d_t \geq 1$, a_t is non-negative since δ_{t-1} is non-negative by the induction hypothesis. Thus m_t is the maximum value of f_t, which implies (2).

For (3), when $r_{K-(t-1)} = r_{K-(t-2)} = \cdots = r_K = b_{K-(t-2)} = \cdots = b_K = 0$, the induction hypothesis gives that $\delta_{t-1} = m_{t-1}$. Since $s_t = 1$ and $d_t = 2(t-1)$ when $r_{K-(t-1)} = b_{K-(t-1)} = 0$, it holds that

$$a_t d_t = m_{t-1}(2s_t + d_t - 1)d_t = m_{t-1}(2t-1)(2t-2) = (2t-1)!,$$

which is exactly m_t. Thus $f_t(0,\ldots,0;0,\ldots,0) = 0$ by (5). Since f_t is monotone by (1), it is minimum.

Therefore, (1)–(3) hold. □

We calculate the function value obtained by an algorithm (Lemma 3.1 (iii)).

Lemma 3.5. *For any $t = 1, 2, \ldots, K$, the constructed function f_t satisfies the following.*

$$f_t(0,\ldots,0,1;1,\ldots,1,0) = t \cdot (2t-2)!.$$

Proof. We prove by induction on t. When $t = 1$, it is straightforward to verify the lemma. By the induction hypothesis, $f_{t-1}(0,\ldots,0,1;1,\ldots,1,0) = (t-1)\cdot(2t-4)!$. Then $\delta_{t-1} = (t-2)\cdot(2t-4)!$. As $s_t = 1$ and $d_t = 2t-3$, it holds by (5) that

$$f_t(0,\ldots,0,1;1,\ldots,1,0) = (2t-1)! - (2\cdot(2t-3)! + (2t-4)!(t-2)(2t-4))\,(2t-3) = t\cdot(2t-2)!,$$

where the last equality is easy to verify. □

We next deal with submodularity of the constructed function.

Lemma 3.6. *For any $t = 1, 2, \ldots, K$, the constructed function f_t satisfies the following: Suppose that $r^1_{K-(t-1)} \geq r^2_{K-(t-1)}, \ldots, r^1_K \geq r^2_K$ and $b^1_{K-(t-1)} \geq b^2_{K-(t-1)}, \ldots, b^1_K \geq b^2_K$. Then*

1. *For any integer i with $K-(t-1) \leq i \leq K$ such that $r^1_i = 0$, it holds that*

$$\begin{aligned} &f_t(r^1_{K-(t-1)},\ldots,r^1_i+1,\ldots,r^1_K;b^1_{K-(t-1)},\ldots,b^1_K) - f_t(r^1_{K-(t-1)},\ldots,r^1_i,\ldots,r^1_K;b^1_{K-(t-1)},\ldots,b^1_K) \\ &\leq f_t(r^2_{K-(t-1)},\ldots,r^2_i+1,\ldots,r^2_K;b^1_{K-(t-1)},\ldots,b^2_K) - f_t(r^2_{K-(t-1)},\ldots,r^2_i,\ldots,r^2_K;b^2_{K-(t-1)},\ldots,b^2_K). \end{aligned} \tag{7}$$

2. *For any integer i with $K-(t-1) \leq i \leq K$, it holds that*

$$\begin{aligned} &f_t(r^1_{K-(t-1)},\ldots,r^1_K;b^1_{K-(t-1)},\ldots,b^1_i+1,\ldots,b^1_K) - f_t(r^1_{K-(t-1)},\ldots,r^1_K;b^1_{K-(t-1)},\ldots,b^1_i,\ldots,b^1_K) \\ &\leq f_t(r^2_{K-(t-1)},\ldots,r^2_K;b^\ell_{K-(t-1)},\ldots,b^2_i+1,\ldots,b^2_K) - f_t(r^2_{K-(t-1)},\ldots,r^2_K;b^2_{K-(t-1)},\ldots,b^2_i,\ldots,b^2_K). \end{aligned} \tag{8}$$

Proof. We proceed by induction on t. The base case $t = 1$ is easy to verify. For the induction step where $t \geq 2$, let d_t^ℓ, s_t^ℓ and δ_{t-1}^ℓ denote the realizations of d_t, s_t, and δ_{t-1} based on $\bigcup_{j=K-(t-1)}^{K}\{r_j^\ell, b_j^\ell\}$ for $\ell = 1, 2$. As $b^1_{K-(t-1)} \geq b^2_{K-(t-1)}$, we have $\hat{b}^1_{K-(t-1)} \geq \hat{b}^2_{K-(t-1)}$, and thus $d_t^1 \leq d_t^2$.

First assume that $i \geq K - (t-2)$. Then it follows from (5) that the left-hand side of (7) is equal to

$$\begin{aligned} d_t^1(d_t^1 - 1)(&f_{t-1}(r^1_{K-(t-2)}, \ldots, r_i^1 + 1, \ldots, r_K^1; b^1_{K-(t-2)}, \ldots, b_K^1) \\ &- f_{t-1}(r^1_{K-(t-2)}, \ldots, r_i^1, \ldots, r_K^1; b^1_{K-(t-2)}, \ldots, b_K^1)). \end{aligned}$$

Similarly, the right-hand side of (7) is equal to

$$\begin{aligned} d_t^2(d_t^2 - 1)(&f_{t-1}(r^2_{K-(t-2)}, \ldots, r_i^2 + 1, \ldots, r_K^2; b^2_{K-(t-2)}, \ldots, b_K^2) \\ &- f_{t-1}(r^2_{K-(t-2)}, \ldots, r_i^2, \ldots, r_K^2; b^2_{K-(t-2)}, \ldots, b_K^2)). \end{aligned}$$

Since $d_t^1(d_t^1 - 1) \leq d_t^2(d_t^2 - 1)$ as $d_t^1 \leq d_t^2$ and both d_t^1 and d_t^2 are non-negative integers, (7) holds by the induction hypothesis. The argument for (8) is identical.

Next assume that $i = K - (t-1)$. Then (7) holds, because the LHS and RHS of (7) are equal to $2m_{t-1}d_t^1$ and $2m_{t-1}d_t^2$, respectively, and $d_t^1 \leq d_t^2$. For (8), first observe that, if $b^1_{K-(t-1)} \geq 2(t-1)$, then $d_t^1 = 0$, implying that the LHS of (8) is zero, and hence (8) is trivial since f_t is monotone by Lemma 3.4. So assume that $b^1_{K-(t-1)} < 2(t-1)$, implying that $d_t^2 \geq d_t^1 > 0$. Furthermore, by the monotonicity, we have $\delta_{t-1}^1 \leq \delta_{t-1}^2$. It follows from (5) that the LHS and RHS of (8) are equal to

$$2m_{t-1}s_t + 2\delta_{t-1}^1(d_t^1 - 1) \text{ and } \ 2m_{t-1}s_t + 2\delta_{t-1}^2(d_t^2 - 1),$$

respectively. Since $d_t^1 \leq d_t^2$, this proves the lemma. □

We are left with proving indistinguishability. We begin by proving that when only elements of a particular class are present, the f_t-values are entirely determined by their cardinality.

Lemma 3.7. *For any $t = 2, 3, \ldots, K$ and any non-negative integer b, it holds that*

$$f_t(1, 0, \ldots, 0; b, 0, \ldots, 0) = f_t(0, 0, \ldots, 0; b+1, 0, \ldots, 0).$$

Proof. Let d_t^1 and d_t^2, δ_{t-1}^1 and δ_{t-1}^2 be the realizations of d_t and δ_{t-1} based on $(1, 0, \ldots, 0; b, 0, \ldots, 0)$ and $(0, 0, \ldots, 0; b+1, 0, \ldots, 0)$, respectively. Then $\delta_{t-1}^1 = \delta_{t-1}^2 = m_{t-1}$ holds by Lemma 3.4. By (5), the two function values stated in the lemma can be written as

$$\begin{aligned} f_t(1, 0, \ldots, 0; b \quad, 0, \ldots, 0) &= m_t - (d_t^1 - 1)d_t^1 m_{t-1}, \\ f_t(0, 0, \ldots, 0; b+1, 0, \ldots, 0) &= m_t - (d_t^2 + 1)d_t^2 m_{t-1}, \end{aligned}$$

respectively. If $b \geq 2(t-1)$, then both d_t^1 and d_t^2 are 0, which gives the equivalence of the two above expressions. So assume that $b + 1 \leq 2(t-1)$, and hence $d_t^1 = d_t^2 + 1$. Then, since

$$(d_t^1 - 1)d_t^1 = (d_t^2 + 1)(d_t^2 + 1 - 1) = (d_t^2 + 1)d_t^2,$$

the above two expressions are equivalent. The proof follows. □

The next lemma generalizes the previous one: if we fix the numbers of red and blue elements of classes before class i, then the function value is entirely determined by the cardinality of elements in class i. This proves Lemma 3.1 (iv) by setting $t = K$.

Lemma 3.8. *For any $i = 2, 3, \dots, K$ and any integer t such that $K - i + 1 \le t \le K$, it holds that*

$$\begin{aligned} &f_t(r_{K-(t-1)}, \dots, r_{i+1}, 1, 0, \dots, 0; b_{K-(t-1)}, \dots, b_{i+1}, b_i, 0, \dots, 0) \\ &= f_t(r_{K-(t-1)}, \dots, r_{i+1}, 0, 0, \dots, 0; b_{K-(t-1)}, \dots, b_{i+1}, b_i + 1, 0, \dots, 0). \end{aligned}$$

Proof. We prove by induction on t. The base case $t = K - i + 1$ follows from Lemma 3.7. For the induction step when $t > K - i + 1$, let δ^ℓ_{t-1} be the realization of δ_{t-1} based on $\bigcup_{j=K-(t-2)}^{i} \{r^\ell_j, b^\ell_j\}$ for $\ell = 1, 2$. Induction hypothesis then states that $\delta^1_{t-1} = \delta^2_{t-1}$ and the proof follows by observing how the function is defined by recurrence (5). □

Lemma 3.1 follows from Lemmas 3.4, 3.5, 3.6, and 3.8.

3.4 Proof of Observations in Section 3.2

We first prove Lemma 3.2.

Proof of Lemma 3.2. Let t be an integer from 2 to K. We first compute $f_t(1, 0, \dots, 0; 0, \dots, 0)$. By Lemma 3.4, $f_{t-1}(0, \dots, 0; 0, \dots, 0) = 0$ and hence $\delta_{t-1} = m_{t-1}$. Hence, since $s_t = 0$ and $d_t = 2(t-1)$, it follows from (5) that

$$f_t(1, 0, \dots, 0; 0, \dots, 0) = m_t - m_{t-1}(2s_t + d_t - 1)d_t = m_t - m_{t-1}(2t-2)(2t-3) = 2 \cdot (2t-2)!.$$

On the other hand, since $s_t = 1$ and $d_t = 2(t-1) - 1$ in the case of $f_t(0, \dots, 0; 1, 0, \dots, 0)$,

$$f_t(0, \dots, 0; 1, 0, \dots, 0) = m_t - m_{t-1}(2s_t + d_t - 1)d_t = m_t - m_{t-1}(2t-2)(2t-3) = 2 \cdot (2t-2)!.$$

Suppose that we are given $r_{K-(t-1)}, r_{K-(t-2)}, \dots, r_K$ and $b_{K-(t-1)}, b_{K-(t-2)}, \dots, b_K$. Let f' be the value of $f_t(r_{K-(t-1)}, r_{K-(t-2)}, \dots, r_K; b_{K-(t-1)}, b_{K-(t-2)}, \dots, b_K)$. Then

$$\begin{aligned} &f_{t+1}(0, r_{K-(t-1)}, r_{K-(t-2)}, \dots, r_K; 0, b_{K-(t-1)}, b_{K-(t-2)}, \dots, b_K) \\ &\quad = m_{t+1} - (2m_t s_{t+1} + (m_t - f')(d_{t+1} - 1))d_{t+1} \\ &\quad = (2t)(2t-1) \cdot f'. \end{aligned}$$

Therefore, when $r_{K-(t-1)} = 1$ and $r_{K-(t-2)} = \cdots = r_K = b_{K-(t-1)} = b_{K-(t-2)} = \cdots = b_K = 0$ ($2 \le t \le K$), it follows that

$$\begin{aligned} f(0, \dots, 0, 1, 0, \dots, 0; 0, \dots, 0) &= (2K-2)(2K-3) \cdots (2t)(2t-1) f_t(1, 0, \dots, 0; 0, \dots, 0) \\ &= (2K-2)(2K-3) \cdots (2t)(2t-1) \, (2 \cdot (2t-2)!) = 2 \cdot (2K-2)!. \end{aligned}$$

Similarly, we have $f(0, \dots, 0; 0, \dots, 0, 1, 0, \dots, 0) = 2 \cdot (2K-2)!$. Moreover, since $f_1(1; 0) = 1$ by the definition, it holds that

$$f(0, \dots, 0, 1; 0, \dots, 0) = (2K-2)(2K-3) \cdots 2 \cdot f_1(1; 0) = (2K-2)!.$$

Thus Lemma 3.2 holds. □

Below we prove Lemma 3.3.

Lemma 3.3. It follows from Lemma 3.4 that

$$f(1,\dots,1;0,\dots,0) = f_K(1,\dots,1;0,\dots,0) = (2K-1)!.$$

Moreover, Lemma 3.3 imlplies that

$$\begin{aligned} &f(1,0,\dots,0;0,\dots,0) + f(0,1,\dots,0;0,\dots,0) + \cdots + f(0,\dots,0,1;0,\dots,0) \\ &= (K-1)\cdot(2\cdot(2K-2)!) + (2K-2)! = (2K-1)!. \end{aligned}$$

Thus the first equality holds.

We next consider computing $f(0,\dots,0;2(K-1),0,\dots,0)$. Since $f_{K-1}(0,\dots,0;0,\dots,0) = 0$, we have $\delta_{K-1} = m_{K-1}$. Hence, since $s_K = 1$ and $d_K = 0$ in this case, we see from (5) that

$$f_K(0,\dots,0;2(K-1),0,\dots,0) = m_K - m_{K-1}(2s_K + (d_K - 1))d_K = m_K.$$

Thus Lemma 3.3 follows. □

4 Algorithms

In this section, we present algorithms for a cardinality constraint and a matroid constraint, respectively.

We first establish some convention here. We call the input problem the *original instance*, in which all elements in the stream are considered. Furthermore,

- K denotes the size constraint for the input cardinality constraint or the rank of the input matroid;
- OPT is the optimal solution of the original instance;
- $f\colon 2^E \to \mathbb{Z}_+$ is the input submodular function.

All algorithms are given in the form of a general procedure with a set of parameters (in particular, a re-defined submodular function and a modified cardinality/matroid constraint). Each invocation of the general procedure is called a *branch*. A branch considers all remaining elements that have not arrived so far (i.e., a suffix of the entire stream of elements). For a branch, we use the following notation.

- k denotes the size constraint for a new cardinality constraint or the rank of the new matroid (as a rule $k \le K$);
- $\overline{\mathrm{OPT}}$ is the intersection of OPT and the remaining elements considered by this branch;
- g is a submodular function derived from f. More specifically, $g(T) = f(T \mid S) := f(T \cup S) - f(S)$, where S is a subset of elements that have already arrived so far before this branch starts.

Whenever $\overline{\mathrm{OPT}}$ is non-empty, we denote by o_1 the first element that arrives in the stream among all elements in $\overline{\mathrm{OPT}}$.

In the description of the algorithms, we assume that we are given an approximate $v \in \mathbb{R}_+$ so that $v \leq f(\overline{\mathrm{OPT}}) \leq (1+\varepsilon)v$. and we will explain in Section 4.3 how to implement them without knowing the value v.

4.1 Cardinality Constraint

We start by defining the main procedure Cardinality. The procedure takes four parameters k, s, v, and g; we assume that $\overline{\mathrm{OPT}}$ satisfies the conditions that $g(\overline{\mathrm{OPT}}) \geq v$ and $|\overline{\mathrm{OPT}}| \leq k$, and s is an upper bound that we impose on the size of the returned solution of Cardinality(k, s, v, f).

Cardinality(k, s, v, g) is described in Algorithm 1. Its basic idea can be simply described as follows. We target the approximation ratio of $\frac{s}{s+k}$, which, intuitively, states that the ratio should get better as the allowed solution size s is increased with respect to k, the upper bound on the size of $\overline{\mathrm{OPT}}$. In case that k or $s = 1$, we simply choose the element that gives the largest g-value. So assume that $k, s \geq 2$.

Depending on the value of $g(o_1)$ (for which we have no prior knowledge), we create two branches:

- **Branch** 1: If $g(o_1)$ is sufficiently large (precisely at least $\frac{v}{k+s-1}$), we just take the first element e so that $g(e) \geq \frac{v}{k+s-1}$. Then e precedes o_1 (or is just o_1) and the rest of $\overline{\mathrm{OPT}}$. We define a new submodular function $g' = g(\cdot \mid e)$. We then invoke Cardinality$(k, s-1, v-g(e), g')$.
- **Branch** 2: If $g(o_1)$ is too small, we simply ignore it and invoke Cardinality$(k-1, s, \frac{k+s-2}{k+s-1}v, g)$ directly.

The output is just the better outcome of the two branches.

Lemma 4.1. *Suppose that $k \geq 1$ and $s \geq 1$. Suppose that there is a set $\overline{\mathrm{OPT}}$ in the input stream so that $g(\overline{\mathrm{OPT}}) \geq v$ and $|\overline{\mathrm{OPT}}| \leq k$. Then Cardinality$(k, s, v, g)$ returns a solution S such that $g(S) \geq \frac{s}{k+s-1}v$ and $|S| \leq s$.*

Proof. We begin by noting our assumption is that $|\overline{\mathrm{OPT}}| \leq k$ and $g(\overline{\mathrm{OPT}}) \geq v$. If $\overline{\mathrm{OPT}} = \emptyset$, then $v \leq f(\emptyset)$, implying that any solution satisfies the lemma. Therefore, in the following, we assume that $\overline{\mathrm{OPT}} \neq \emptyset$.

We now prove by induction, first on k and then on s. In the base case $k = 1$, as the algorithm chooses an element e maximizing $g(e)$, we have $g(e) \geq g(o_1) = g(\overline{\mathrm{OPT}}) \geq v$.

For the induction step $k > 1$, we apply induction on s. In the base case $s = 1$, again the algorithm chooses an element e maximizing $g(e)$, so $g(e) \geq g(o_1) \geq \frac{g(\overline{\mathrm{OPT}})}{k} \geq \frac{v}{k}$, where the second inequality follows from submodularity. For the induction step $s > 1$, the algorithm creates two branches. Now consider two possibilities.

- Suppose that $g(o_1) \geq \frac{v}{k+s-1}$. Then **Branch** 1 is bound to find an element e with $g(e) \geq \frac{v}{k+s-1}$ and e either precedes o_1 or is just o_1. As $g'(\overline{\mathrm{OPT}}) \geq g(\overline{\mathrm{OPT}}) - g(e) \geq v - g(e)$, we can then

Algorithm 1

procedure Cardinality(k, s, v, g)
  if $k \geq 2$ and $s \geq 2$ **then**
    (Branch 1)
      Let e be the first element such that $g(e) \geq \frac{v}{k+s-1}$.
      Define $g' = g(\cdot \mid e)$.
      Apply Cardinality$(k, s-1, v-g(e), g')$ on all the elements after e.
      Let S' be the returned solution.
      $S_1 := S' + e$.
    (Branch 2)
      Apply Cardinality$(k-1, s, \frac{k+s-2}{k+s-1}v, g)$ on all the elements.
      Let S_2 be the returned solution.
    if $g(S_1) > g(S_2)$ **then return** S_1.
    else return S_2.
  if $k = 1$ or $s = 1$ **then return** $\arg\max_e g(e)$.

apply induction hypothesis on Cardinality$(k, s-1, v-g(e), g')$, which returns a solution S' with $g'(S') \geq \frac{s-1}{k+s-2}\big(v - g(e)\big)$ and $|S'| \leq s-1$. Then

$$g(S_1) = g'(S') + g(e) \geq \frac{s-1}{k+s-2}v + \frac{k-1}{k+s-2}g(e) \geq \frac{s}{k+s-1}v.$$

Clearly, $|S_1| \leq s$.

- Suppose that $g(o_1) < \frac{v}{k+s-1}$. Then $g(\overline{\mathrm{OPT}} - o_1) \geq g(\overline{\mathrm{OPT}}) - g(o_1) \geq \frac{k+s-2}{k+s-1}v$, due to submodularity. By the induction hypothesis, Cardinality$(k-1, s, \frac{k+s-2}{k+s-1}v, g)$ in **Branch** 2 returns a solution S_2 with $g(S_2) \geq \frac{s}{k+s-2}\frac{k+s-2}{k+s-1}v = \frac{s}{k+s-1}v$. Clearly, $|S_2| \leq s$.

Therefore, one of the two branches gives the desired solution. This finishes the induction step on s and then also on k. The proof follows. □

Theorem 4.2. *Suppose that $v \leq f(\mathrm{OPT})$. Then,the algorithm Cardinality(K, K, v, f) returns a solution S with $f(S) \geq \frac{K}{2K-1}v$. The space complexity (for a fixed v) is $O\left(K2^{2K}\right)$.*

Proof. The first part follows from Lemma 4.1. For space requirement, let $\Gamma(k, s)$ denotes the space required for Cardinality(k, s, v, g) and let c be some constant. Then it follows from the algorithm that (1) $\Gamma(k, s) \leq c$ when $k = 1$ or $s = 1$, and (2) $\Gamma(k, s) \leq \Gamma(k-1, s) + \Gamma(k, s-1) + c$ when $k > 1$ and $s > 1$. It is easy to verify this recurrence leads to $\Gamma(k, s) \leq k2^{k+s}c$. The proof follows. □

Therefore, if we are given v such that $v \leq f(\mathrm{OPT}) \leq (1+\varepsilon)v$, Cardinality$(K, K, v, f)$ returns a solution S with $f(S) \geq \left(\frac{K}{2K-1} - \varepsilon\right) f(\mathrm{OPT})$. Our algorithm consists in invoking Cardinality(K, K, v, f) for v in some interval, while updating the interval dynamically. See Section 4.3 for details.

Algorithm 2

procedure Matroid(k, v, g, I)
  if $k > 1$ **then**
    $\beta := \max_{b \in \mathbb{Z}_+} \frac{b}{K^4} \le \frac{v}{2}$.
    for $0 \le b \le \beta$ **do**
      $T := \emptyset$.
      while $|I \cup T| < K$ **do**
        Let e be the first element in the remaining stream satisfying (1) $I \cup T + e \in \mathcal{I}$, and (2) $g(e) \ge \frac{bv}{K^4}$.
        Branch $(b, |T| + 1)$
          Define $g' = g(\cdot \mid e)$.
          Apply Matroid$(k - 1, (1 - \frac{1}{K^4})v - 2g(e), g', I + e)$ on all the elements after e.
          Let the returned solution be S'.
          $S_{b,|T|+1} := e + S'$.
        $T := T + e$.
  Branch 0
    $S_0 := \arg\max_{e, e \in \mathcal{I}} g(e)$.
  return $\arg\max \{g(S) \mid S \in \{S_{b,j} \mid b = 0, \ldots, \beta, j = 1, \ldots, k\} \cup S_0\}$.

4.2 Matroid Constraint

Let $\mathcal{M} = (E, \mathcal{I})$ be a given matroid, whose ground set E is the entire stream of elements. Assume that the rank of $\mathcal{M}$ is K. We present the procedure Matroid(k, v, g, I) as Algorithm 2. Again g is the submodular function defined over the remaining elements, among which $\overline{\text{OPT}}$ satisfies the conditions that $g(\overline{\text{OPT}}) \ge v$ and $|\overline{\text{OPT}}| = k \le K$. Moreover, an independent set $I \in \mathcal{I}$ is also given as a part of the input; such a set I should guarantee that $I \cup \overline{\text{OPT}} \in \mathcal{I}$.

We now give the intuition of this procedure. We guess out possible values of $g(o_1)$ in intervals of $\frac{v}{K^4}$ (between 0 and $0.5v$). For each possible interval $\left[\frac{bv}{K^4}, \frac{(b+1)v}{K^4}\right]$ of $g(o_1)$, we create a branch, and we could potentially take the first element e so that $I + e \in \mathcal{I}$, $g(e) \ge \frac{bv}{K^4}$, and then define $g' = g(\cdot \mid e)$ and invoke Matroid$(k - 1, (1 - \frac{1}{K^4})v - 2g(e), g', I + e)$ on the elements after e. So far the idea is similar to the previous cardinality case. The problem of this approach is that there is no guarantee that $(I + e) \cup (\overline{\text{OPT}} - o_1) \in \mathcal{I}$, that is a required condition for the procedure Matroid$(k - 1, (1 - \frac{1}{K^4})v - 2g(e), g', I + e)$.

To remedy this issue, we introduce the following idea. For each possible interval $\left[\frac{bv}{K^4}, \frac{(b+1)v}{K^4}\right]$ of $g(o_1)$, we create a set T, initialized as $\emptyset$. Every time a new element e arrives so that (1) $I \cup T + e \in \mathcal{I}$, and (2) $g(e) \ge \frac{bv}{K^4}$, we add e into T and create a new branch Matroid$(k-1, (1-\frac{1}{K^4})v-2g(e), g', I+e)$, where $g' = g(\cdot \mid e)$. As we will show (see the proof of Lemma 4.5), at least one of the elements $e \in T$ satisfies the property $(I + e) \cup (\overline{\text{OPT}} - o_1) \in \mathcal{I}$. Apparently, $|T| \le K$, so in total we have at most $K + 1$ branches for each b.

The following fact is well-known.

Proposition 4.3. *Suppose that C_1 and C_2 are two circuits and $x \in C_1 \cap C_2$. Then, for any $y \in C_1 \backslash C_2$, there exists another circuit $C \subseteq C_1 \cup C_2 - x$ and $C \ni y$.*

Lemma 4.4. *Let $T = \{e_1, \ldots, e_{|T|}\}$. Suppose that $I \cup T \in \mathcal{I}$ and $I \cup \overline{\mathrm{OPT}} \in \mathcal{I}$ holds. Furthermore, suppose that for each $e_i \in T$, $I \cup \overline{\mathrm{OPT}} + e_i$ contains a circuit C_i so that $C_i \not\ni o_1$. Then $I \cup T + o_1 \in \mathcal{I}$.*

Proof. We prove by establishing a more general statement: there is no circuit $C^* \subseteq I \cup \overline{\mathrm{OPT}} \cup T$ so that $C^* \ni o_1$ (this implies that the lemma as $I \cup T \in \mathcal{I}$). We proceed by contradiction. Assume that such a circuit C^* does exist and we will establish the following claim.

Claim 4. *Suppose that $C^* \ni o_1$ and $C^* \subseteq I \cup \overline{\mathrm{OPT}} \cup T$ and $C^* \cap T \neq \emptyset$. Then, there exists another circuit $\overline{C} \subseteq I \cup \overline{\mathrm{OPT}} \cup T$, $\overline{C} \ni o_1$ and $|\overline{C} \cap T| < |C^* \cap T|$.*

Proof. To prove the claim, assume that $e_i \in C^* \cap T$. Then by Proposition 4.3, we have a circuit $\overline{C} \subseteq C_i \cup C^* - e_i$ and $\overline{C} \ni o_1$ (recall that C_i is the circuit contained in $I \cup \overline{\mathrm{OPT}} + e_i$ and $C_i \not\ni o_1$). Clearly, $|\overline{C} \cap T| < |C^* \cap T|$. □

Now by this claim, we conclude that there is a circuit $\overline{C} \subseteq I \cup \overline{\mathrm{OPT}}$ and $\overline{C} \ni o_1$, a contradiction to the assumption that $I \cup \overline{\mathrm{OPT}} \in \mathcal{I}$. □

Lemma 4.5. *Suppose that there is a set $\overline{\mathrm{OPT}}$ in the input stream so that $g(\overline{\mathrm{OPT}}) \geq v$, $I \cup \overline{\mathrm{OPT}} \in \mathcal{I}$, and $|\overline{\mathrm{OPT}}| = k$. Then $\mathsf{Matroid}(k, v, g, I)$ returns a solution S so that $g(S) \geq \frac{1}{2}\left(1 - \frac{1}{2K-k}\right)v$ and $I \cup S \in \mathcal{I}$.*

Proof. We prove by induction on k. For the base case $k = 1$, **Branch** 0 is bound to get an element e so that $g(e) \geq g(\overline{\mathrm{OPT}}) \geq v$. So assume that $k > 1$. Then there exists b with $0 \leq b \leq \beta$ such that $\frac{bv}{K^4} \leq g(o_1) \leq \frac{(b+1)v}{K^4}$. Let $T = \{e_1, \ldots, e_{|T|}\}$ be the set collected for this b at the moment immediately before o_1 arrives. There are two possibilities.

- If some element $e_i \in T$ satisfies the condition that either $I \cup \overline{\mathrm{OPT}} + e_i$ is independent or contains a circuit C_i and $C_i \ni o_1$, then $I \cup \overline{\mathrm{OPT}} + e_i - o_1 \in \mathcal{I}$. Moreover, it holds that $g'(\overline{\mathrm{OPT}} - o_1) \geq g(\overline{\mathrm{OPT}}) - g(o_1) - g(e) \geq \left(1 - \frac{1}{K^4}\right)v - 2g(e)$ as $g(o_1) \leq g(e) + \frac{v}{K^4}$.
- Otherwise, that is, if every element $e_i \in T$ satisfies the condition that $I \cup \overline{\mathrm{OPT}} + e_i$ contains a circuit C_i and $C_i \not\ni o_1$, then by Lemma 4.4, $I \cup T + o_1 \in \mathcal{I}$, implying that o_1 will be added into T.

In both cases, we know that there exists an element $e \in T \cup \{o_1\}$ so that (1) $g(e) \geq \frac{bv}{K^4}$, (2) $g'(\overline{\mathrm{OPT}} - e) \geq \left(1 - \frac{1}{K^4}\right)v - 2g(e)$, and (3) $(I + e) \cup (\overline{\mathrm{OPT}} - o_1) \in \mathcal{I}$. Furthermore, all elements of $\overline{\mathrm{OPT}} - o_1$ are considered by $\mathsf{Matroid}(k-1, \left(1 - \frac{1}{K^4}\right)v - 2g(e), g', I + e)$. Then by induction hypothesis, in **Branch**(b, j) for some j, we obtain a solution

$$\begin{aligned} g(S_{b,j}) &\geq g(e) + \left(\left(1 - \frac{1}{K^4}\right)v - 2g(e)\right)\frac{1}{2}\left(1 - \frac{1}{2K-k+1}\right) \\ &\geq \frac{1}{2}\left(1 - \frac{1}{2K-k}\right)v. \end{aligned}$$

This finishes the induction step.

The fact that the returned solution S satisfies $S \cup I \in \mathcal{I}$ is easy to verify. □

Theorem 4.6. *Suppose that $v \le f(\mathrm{OPT})$. Then the algorithm $\mathsf{Matroid}(K, v, f, \emptyset)$ returns a solution S guaranteeing that $f(S) \ge \left(\frac{1}{2} - \frac{1}{2K}\right) v$. The space complexity (for a fixed v) is $O\left(K^{5K+1}\right)$.*

Proof. The first part follows from Lemma 4.5. For the space complexity, consider the branching tree with $\mathsf{Matroid}(K, v, f, \emptyset)$ as the root. By the algorithm, each node has at most $O(K^5)$ branches. Furthermore, the space required for such a node is $K + c$ for some constant c (for each branch, we need to store an element). The depth of such a tree is at most K. So the total complexity is at most $O(K^{5K+1})$. □

4.3 Implementation

We now explain how to implement the algorithms described in the preceding sections without knowing the optimal value v. We adapt the dynamic-update technique in [2].

We here explain the cardinality case. The matroid case is analogous. We will let v be a number of the form $\left(1+\varepsilon\right)^i$ for some $i \in \mathbb{Z}$. We observe that $\max_e f(e) \le f(\mathrm{OPT}) \le K \max_e f(e)$. Let m be the maximum of $f(e)$ among all elements e that have arrived so far. The algorithm $\mathsf{Cardinality}(K, K, v, f)$ is activated only when $\frac{m}{(1+\varepsilon)^2} \le v \le \frac{Km}{\varepsilon}$. The critical observation is that, if $f(\mathrm{OPT}) > \frac{Km}{\varepsilon}$, then the set of optimal items $\mathrm{OPT}' \setminus \mathrm{OPT}$ that have arrived so far has the property that

$$f(\mathrm{OPT}') \le \sum_{o_i \in \mathrm{OPT}'} f(o_i) \le Km \le \varepsilon f(\mathrm{OPT}).$$

Therefore, the first time an element e arrives so that $m := f(e)$ and $m \le f(\mathrm{OPT}) \le \frac{Km}{\varepsilon}$, there exists a subset $\overline{\mathrm{OPT}} = \mathrm{OPT} \backslash \mathrm{OPT}'$ so that $f(\overline{\mathrm{OPT}}) \ge (1-\varepsilon) f(\mathrm{OPT})$. This means that we can use $\overline{\mathrm{OPT}}$ instead of OPT to perform the algorithm. Then the interval $\left[\frac{m}{(1+\varepsilon)^2}, \frac{Km}{\varepsilon}\right]$ contains v such that $v \le f(\overline{\mathrm{OPT}}) \le (1+\varepsilon)v$, since $m \le f(\mathrm{OPT}) \le \frac{Km}{\varepsilon}$. It follows from Theorem 4.2 that, using $O\left(K 2^{2K}\right)$ space, $\mathsf{Cardinality}(K, K, v, f)$ returns a solution S with $f(S) \ge \frac{K}{2K-1} v$, implying that

$$f(S) \ge \frac{K}{2K-1}(1-\varepsilon) f(\overline{\mathrm{OPT}}) \ge \frac{K}{2K-1}(1 - O(\varepsilon)) f(\mathrm{OPT}).$$

The number of guesses for v is equal to $O\left(\log_{1+\varepsilon}\left(\frac{K}{\varepsilon}\right)\right) = O\left(\frac{\log(K/\varepsilon)}{\varepsilon}\right)$.

Theorems 1.4 and 1.5 follow from the preceding discussion and Theorems 4.2 and 4.6.

References

[1] N. Alon, I. Gamzu, and M. Tennenholtz. Optimizing budget allocation among channels and influencers. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*, pages 381–388, 2012.

[2] A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause. Streaming submodular maximization: massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 671–680, 2014.

[3] A. Badanidiyuru and J. Vondrák. Fast algorithms for maximizing submodular functions. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1497–1514, 2013.

[4] E. Balkanski, A. Rubinstein, and Y. Singer. An exponential speedup in parallel running time for submodular maximization without loss in approximation. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 283–302, 2019.

[5] E. Balkanski and Y. Singer. The adaptive complexity of maximizing a submodular function. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1138–1151, 2018.

[6] R. Barbosa, A. Ene, H. L. Nguyễn, and J. Ward. The power of randomization: Distributed submodular maximization on massive datasets. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning (ICML)*, pages 1236–1244, 2015.

[7] R. D. P. Barbosa, A. Ene, H. L. Nguyễn, and J. Ward. A new framework for distributed submodular maximization. In *Proceedings of the IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 645–654, 2016.

[8] M. Bateni, H. Esfandiari, and V. Mirrokni. Almost optimal streaming algorithms for coverage problems. In *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 13–23, 2017.

[9] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[10] A. Chakrabarti and S. Kale. Submodular maximization meets streaming: matchings, matroids, and more. *Mathematical Programming*, 154(1-2):225–247, 2015.

[11] T.-H. H. Chan, Z. Huang, S. H.-C. Jiang, N. Kang, and Z. G. Tang. Online submodular maximization with free disposal: Randomization beats for partition matroids online. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1204–1223, 2017.

[12] T.-H. H. Chan, S. H.-C. Jiang, Z. G. Tang, and X. Wu. Online submodular maximization problem with vector packing constraint. In *Proceedings of the 25th Annual European Symposium on Algorithms (ESA)*, pages 24:1–24:14, 2017.

[13] C. Chekuri, S. Gupta, and K. Quanrud. Streaming algorithms for submodular function maximization. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 318–330, 2015.

[14] C. Chekuri and K. Quanrud. Submodular function maximization in parallel via the multilinear relaxation. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 303–322, 2019.

[15] C. Chekuri, J. Vondrák, and R. Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM Journal on Computing*, 43(6):1831–1879, 2014.

[16] A. Ene and H. L. Nguyễn. A nearly-linear time algorithm for submodular maximization with a knapsack constraint. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 132, pages 53:1–53:12, 2019.

[17] A. Ene and H. L. Nguyễn. Submodular maximization with nearly-optimal approximation and adaptivity in nearly-linear time. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 274–282, 2019.

[18] A. Ene and H. L. Nguyễn. Towards nearly-linear time algorithms for submodular maximization with a matroid constraint. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 132, pages 54:1–54:14, 2019.

[19] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[20] Y. Filmus and J. Ward. A tight combinatorial algorithm for submodular maximization subject to a matroid constraint. *SIAM Journal on Computing*, 43(2):514–542, 2014.

[21] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming*, pages 265–294, 1978.

[22] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions II. *Mathematical Programming Study*, 8:73–87, 1978.

[23] C.-C. Huang and N. Kakimura. Multi-pass streaming algorithms for monotone submodular function maximization. *CoRR*, abs/1802.06212, 2018.

[24] C.-C. Huang and N. Kakimura. Improved streaming algorithms for maximizing monotone submodular functions under a knapsack constraint. In *Proceedings of the Algorithms and Data Structures Symposium (WADS)*, pages 438–451, 2019.

[25] C.-C. Huang, N. Kakimura, and Y. Yoshida. Streaming algorithms for maximizing monotone submodular functions under a knapsack constraint. *Algorithmica*, 2019.

[26] E. Kazemi, M. Mitrovic, M. Zadimoghaddam, S. Lattanzi, and A. Karbasi. Submodular streaming in all its glory: Tight approximation, minimum memory and low adaptive complexity. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 3311–3320, 2019.

[27] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 137–146, 2003.

[28] A. Krause, A. P. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

[29] A. Kulik, H. Shachnai, and T. Tamir. Maximizing submodular set functions subject to multiple linear constraints. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 545–554, 2013.

[30] R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani. Fast greedy algorithms in MapReduce and streaming. *ACM Transactions on Parallel Computing*, 2(3):14:1–14:22, 2015.

[31] J. Lee. *Maximum Entropy Sampling*, volume 3 of *Encyclopedia of Environmetrics*, pages 1229–1234. John Wiley & Sons, Ltd., 2006.

[32] J. Lee, M. Sviridenko, and J. Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Mathematics of Operations Research*, 35(4):795–806, 2010.

[33] H. Lin and J. Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 912–920, 2010.

[34] H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 510–520, 2011.

[35] A. McGregor and H. T. Vu. Better streaming algorithms for the maximum coverage problem. *Theory of Computing Systems*, 63(7):1595, 2019.

[36] G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.

[37] A. Norouzi-Fard, J. Tarnawski, S. Mitrovic, A. Zandieh, A. Mousavifar, and O. Svensson. Beyond 1/2-approximation for submodular maximization on massive data streams. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 3826–3835, 2018.

[38] T. Soma, N. Kakimura, K. Inaba, and K. Kawarabayashi. Optimal budget allocation: Theoretical guarantee and efficient algorithm. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 351–359, 2014.

[39] J. Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM Journal on Computing*, 42(1):265–304, 2013.

[40] L. Wolsey. Maximising real-valued submodular functions: primal and dual heuristics for location problems. *Mathematics of Operations Research*, 1982.

[41] Y. Yoshida. Maximizing a monotone submodular function with a bounded curvature under a knapsack constraint. *SIAM Journal on Discrete Mathematics*, 33(3):1452–1471, 2018.

[42] Q. Yu, E. L. Xu, and S. Cui. Streaming algorithms for news and scientific literature recommendation: Submodular maximization with a d-knapsack constraint. *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1295–1299, 2016.